

## PREFACIO

*Esta tesis es presentada como parte de los requisitos para optar al grado Académico de Doctor en Ciencias de la Computación, de la Universidad Nacional del Sur y no ha sido presentada previamente para la obtención de otro título en esta Universidad u otras. La misma contiene los resultados obtenidos en investigaciones llevadas a cabo en la Planta Piloto de Ingeniería Química (UNS-CONICET) durante el período comprendido entre el 29 de agosto de 1996 y el 20 de diciembre de 2000, bajo la dirección del Dr. Guillermo Ricardo Simari, profesor titular del Departamento de Ciencias de la Computación de la Universidad Nacional del Sur, y la codirección de la Dra. Nélica Beatriz Brignole, profesora adjunta del Departamento de Ciencias de la Computación de la Universidad Nacional del Sur e investigadora del CONICET.*

*Deseo expresar en primer término mi reconocimiento al Dr. Guillermo Ricardo Simari, por su apoyo y colaboración a lo largo de todos estos años de trabajo.*

*Mi más profundo agradecimiento a la Dra. Nélica Beatriz Brignole, por su invaluable colaboración y espíritu de trabajo, por la formación brindada y por su estímulo constante en la concreción de las investigaciones, los cuales resultaron fundamentales en el desarrollo y concreción de esta tesis.*

*Al Consejo Nacional de Investigaciones Científicas y Técnicas, a la Planta Piloto de Ingeniería Química y al Departamento de Ciencias de la Computación, mi agradecimiento por haberme facilitado los medios para realizar esta tarea.*

*Mi especial reconocimiento a la Dra. Mabel Cristina Sánchez, por su valiosa colaboración sin la cual habría resultado imposible llevar adelante estas investigaciones.*

*A mi amigo y colega Lic. Gustavo Esteban Vazquez, cuya amistad y ayuda resultó fundamental a lo largo de estos años de trabajo en el PLAPIQUI.*

*A la Ing. Susana Espinoza y al Ing. Anibal Blanco, quienes con su constante apoyo y camaradería crearon un ámbito de trabajo excepcional.*

*Y por último, quiero expresar mi mayor gratitud a mis padres, quienes con su afecto y aliento hicieron posible que concretaré esta tesis. A ellos mi mayor agradecimiento por todo el cariño y apoyo brindado.*

*Ignacio Ponzoni*

*Bahía Blanca, 29 de marzo de 2001*

## RESUMEN

El objetivo de esta tesis ha sido diseñar nuevos algoritmos en el campo del análisis de observabilidad de procesos industriales empleando teoría de grafos y conceptos avanzados de ciencias de la computación. Como resultado de estas investigaciones se ha logrado el desarrollo de técnicas robustas y eficientes especialmente diseñadas para la clasificación de variables no medidas en procesos industriales con modelos matemáticos fuertemente no lineales.

Mediante el empleo de los nuevos algoritmos propuestos en esta tesis ahora es posible el tratamiento en forma precisa y eficiente de problemas que no podían ser resueltos por los métodos de observabilidad clásicos, o que requerían una estricta simplificación de su modelo matemático para que estas técnicas pudieran ser aplicadas.

Los métodos desarrollados se basan fundamentalmente en la permutación de la matriz de ocurrencia correspondiente al sistema de ecuaciones que modela la planta. Estos reordenamientos estructurales emplean técnicas de descomposición de grafos, digrafos, bigrafos e hipergrafos. Todas las técnicas desarrolladas lograron un muy buen desempeño, respecto de las metodologías existentes, al ser empleadas en la clasificación de variables no medidas de modelos matemáticos complejos correspondientes a problemas industriales reales.

Finalmente, se diseñó e implementó un sistema de soporte de decisión que engloba toda la experiencia adquirida en clasificación de variables a lo largo de este trabajo de tesis. El software desarrollado resulta eficiente, robusto y amigable, asistiendo al usuario en forma versátil y confiable en la compleja tarea de establecer la ubicación más apropiada para los sensores que controlan el correcto funcionamiento de una planta real. El paquete posibilita analizar en forma rigurosa plantas de cualquier dimensión, incluso las de gran envergadura.

## TABLA DE CONTENIDOS

<i>PREFACIO</i>	<i>1</i>
<i>RESUMEN</i>	<i>3</i>
<i>TABLA DE CONTENIDOS</i>	<i>4</i>
<i>LISTA DE PUBLICACIONES</i>	<i>8</i>
<b><u>1. INTRODUCCIÓN</u></b>	<b><u>11</u></b>
<i>1.1. OBJETIVOS Y ALCANCE DE LA TESIS</i>	<i>14</i>
<i>1.2. CONTENIDOS Y ESTRUCTURA DE LA TESIS</i>	<i>16</i>
<b><u>2. CONCEPTOS BÁSICOS DE TEORÍA DE GRAFOS</u></b>	<b><u>17</u></b>
<i>2.1. INTRODUCCION</i>	<i>17</i>
<i>2.2. GRAFOS, DIGRAFOS Y NOCIONES DE CONECTIVIDAD</i>	<i>19</i>
<i>2.3. BÚSQUEDAS EN PROFUNDIDAD SOBRE GRAFOS</i>	<i>22</i>
2.3.1. ALGORITMOS BÁSICOS	23
2.3.2. DETECCIÓN DE LAS COMPONENTES FUERTES DE UN DIGRAFO USANDO DFS	25
<i>2.4. GRAFOS BIPARTITOS Y PAREAMIENTOS MAXIMALES</i>	<i>29</i>
<i>2.5. GRAFOS Y PERMUTACIÓN DE MATRICES</i>	<i>32</i>
2.5.1. MATRICES DE ADYACENCIA	32
2.5.2. PERMUTACIÓN DE MATRICES CUADRADAS A FORMA TRIANGULAR EN BLOQUES	33
2.5.3. PERMUTACIÓN DE MATRICES NO CUADRADAS A FORMA TRIANGULAR EN BLOQUES	37
<i>2.6. HIPERGRAFOS</i>	<i>39</i>

<b>3. CLASIFICACIÓN DE VARIABLES EN PROCESOS INDUSTRIALES</b>	<b>41</b>
<i>3.1. INTRODUCCIÓN</i>	<i>41</i>
<i>3.2. METODOLOGÍAS DE CLASIFICACIÓN</i>	<i>45</i>
3.2.1. ENFOQUE ORIENTADO A TOPOLOGÍA	46
3.2.2. ENFOQUE ORIENTADO A ECUACIONES	49
3.2.2.1. Técnicas no estructurales	49
3.2.2.2. Técnicas estructurales	51
<b>4. GS-FLCN: OBSERVABILIDAD Y GRAFOS NO DIRIGIDOS</b>	<b>55</b>
<i>4.1. INTRODUCCIÓN</i>	<i>55</i>
<i>4.2. ASPECTOS BÁSICOS DEL GS-FLCN</i>	<i>56</i>
4.2.1. ESTRATEGIA GLOBAL DE BÚSQUEDA DE SUBCONJUNTOS DE ASIGNACIÓN	56
4.2.2. PRIMER NODO MENOS CONECTADO (FIRST LEAST-CONNECTED NODE)	62
<i>4.3. COMPARACIÓN TEÓRICA ENTRE DCB Y GS-FLCN</i>	<i>65</i>
<i>4.4. DCB VERSUS GS-FLCN: APLICACIONES INDUSTRIALES</i>	<i>68</i>
4.4.1. CASO DE ESTUDIO I: EJEMPLO ACADÉMICO	68
4.4.2. CASO DE ESTUDIO II: PLANTA INDUSTRIAL DE PRODUCCIÓN DE ETANO.	69
4.4.2.1. Descripción de la Planta	69
4.4.2.2. Factores de ramificación	71
4.4.3. ANÁLISIS DE LOS RESULTADOS	72
4.4.4. TEST DE ADMISIÓN EMPLEADO EN GS-FLCN	73
<i>4.5. GS-pFLCN: UN ALGORITMO PARALELO-DISTRIBUIDO</i>	<i>76</i>
4.5.1. PARALELIZACIÓN DE GS-FLCN	76
4.5.2. ANÁLISIS DEL DESPEMPEÑO DE GS-pFLCN	78
4.5.2.1. Casos de estudio industriales	78
4.5.2.2. Resultados del análisis de observabilidad efectuado con GS-pFLCN	79
4.5.2.3. Ganancias en términos de robustez	80
<i>4.6. CONCLUSIONES</i>	<i>81</i>

<b>5. CDHG: OBSERVABILIDAD BASADA EN HIPERGRAFOS</b>	<b>84</b>
<i>5.1. INTRODUCCIÓN</i>	<b>84</b>
<i>5.2. CDHG: UN ALGORITMO BASADO EN HIPERGRAFOS</i>	<b>85</b>
5.2.1. ALGORITMO BÁSICO	85
5.2.2. HEURÍSTICAS	91
<i>5.3. ANÁLISIS DEL DESEMPEÑO DE CDHG</i>	<b>93</b>
5.3.1. RESULTADOS	93
<i>5.4. ENFOQUE HÍBRIDO</i>	<b>95</b>
5.4.1. ALGORITMO BÁSICO	95
5.4.2. UNA APLICACIÓN INDUSTRIAL	96
<i>5.5. CONCLUSIONES</i>	<b>98</b>
<b>6. MÉTODO DIRECTO: OBSERVABILIDAD, BIGRAFOS Y DIGRAFOS</b>	<b>100</b>
<i>6.1. INTRODUCCIÓN</i>	<b>100</b>
<i>6.2. MÉTODO DIRECTO PARA ANÁLISIS DE OBSERVABILIDAD</i>	<b>102</b>
6.2.1. ESTRUCTURA GENERAL DEL MÉTODO	102
6.2.2. DESCOMPOSICIÓN GRUESA	105
6.2.3. DESCOMPOSICIÓN FINA	107
6.2.4. TEST DE ADMISIÓN, REASIGNACIÓN DE BLOQUES Y REDUCCIÓN DEL BIGRAFO	109
<i>6.3. ANÁLISIS DE DESEMPEÑO DEL MÉTODO DIRECTO</i>	<b>114</b>
6.3.1. CASO DE ESTUDIO I: COLUMNA DE DESTILACIÓN.	114
6.3.2. CASO DE ESTUDIO II: PLANTA DE SÍNTESIS DE AMONÍACO.	116
6.3.3. CASO DE ESTUDIO III: PLANTA DE PRODUCCIÓN DE ETANO.	118
<i>6.4. COMENTARIOS FINALES</i>	<b>121</b>

<b>7. SISTEMA DE SOPORTE DE DECISIÓN PARA OBSERVABILIDAD</b>	<b>124</b>
<i>7.1. SISTEMAS DE SOPORTE DE DECISIÓN</i>	<i>124</i>
7.1.1. DSSs EN INGENIERÍA	126
<i>7.2. DISEÑO DEL DSS</i>	<i>127</i>
7.2.1. ESTRUCTURA GENERAL DEL DSS	128
7.2.2. INTERFACE: MÓDULO DE INGRESO DE DATOS (MID)	128
7.2.3. MÓDULO DE GENERACIÓN DE MODELOS (MGM)	131
7.2.4. MÓDULO DE ANÁLISIS DE OBSERVABILIDAD (MAO)	132
7.2.5. MÓDULO DE SOPORTE DE DECISIÓN (MSD)	133
<i>7.3. UN CASO DE ESTUDIO INDUSTRIAL</i>	<i>136</i>
7.3.1. CLASIFICACIÓN INICIAL	136
7.3.2. ETAPA DE TOMA DE DECISIONES	137
<i>7.4. COMENTARIOS FINALES</i>	<i>138</i>
<b>8. CONCLUSIONES E INVESTIGACIONES FUTURAS</b>	<b>140</b>
<i>8.1. CONCLUSIONES</i>	<i>141</i>
<i>8.2. INVESTIGACIONES FUTURAS</i>	<i>146</i>
<i>APÉNDICE: MODELADO DE PLANTAS QUÍMICAS USANDO EL DSS</i>	<i>149</i>
<i>REFERENCIAS</i>	<i>171</i>
<i>ÍNDICE DE ALGORITMOS</i>	<i>179</i>
<i>ÍNDICE DE FIGURAS</i>	<i>180</i>
<i>ÍNDICE DE TABLAS</i>	<i>182</i>
<i>ÍNDICE TEMÁTICO</i>	<i>183</i>

## LISTA DE PUBLICACIONES

En Revistas Científicas y Libros con Referato.

Ponzoni I., Vazquez G.E., Sánchez M.C., Brignole N.B. "Parallel Observability Analysis on Networks of Workstations", Computers & Chemical Engineering. Vol 25, No.7-8, 997-1002 (2001). ISSN: 0098-1354.

Vazquez G.E., Ponzoni I., Sánchez M.C., Brignole N.B. "ModGen: A Model Generator for Instrumentation Analysis", Advances in Engineering Software. Including Computing Systems in Engineering. Vol. 32, No. 1, 37-48 (2001). ISSN: 0965-9978.

Ponzoni I., Vazquez G.E., Sánchez M.C., Brignole N.B. "A Computer-Aided DSS for Observability Analysis"<sup>1</sup>, Signal Processing, Communications and Computer Science, Electrical and Computer Engineering Series, 222-227 (2000). ISBN: 960-8052-18-1.

Ponzoni I., Sánchez M.C., Brignole N.B. "A New Structural Algorithm for Observability Classification", Industrial & Engineering Chemistry Research, Vol. 38, No. 8, 3027-3035 (1999). ISSN: 0888-5885.

Ponzoni I., Sánchez M.C., Brignole N.B. "CDHG: a New Partitioning Algorithm based on the Detection of Cycles in Hypergraphs", Latin American Applied Research. Vol. 28, N°1/2, 31-36 (1998). ISSN: 0327-0793.

Ponzoni I., Sánchez M.C., Brignole N.B. "Permutation of Sparse Matrices to a specific lower BTF using Graph Decompositions", Electronical Journal of the Argentine Society for Informatics and Operations Research. Vol. 1, N° 1, 76-87, (1998). ISSN: 1514-6674.

Link: <http://www.dc.uba.ar/sadio/ejs>.

En Revistas Nacionales que contienen una Selección de Trabajos a Congresos.

Ponzoni I., Sánchez M.C., Brignole N.B. "An Efficient Implementation of First Least-Connected Node Algorithm", Mecánica Computacional, Vol. XVIII, pp. 629-638, 1997.

Ponzoni I., Sánchez M.C., Brignole N.B. "Algoritmos eficientes en problemas de asignación", Mecánica Computacional, Vol. XV, pp. 415-424., 1995.

En Congresos Internacionales con Comité de Revisión.

Ferraro, S., Ponzoni, I., Sánchez, M.C., Brignole, N.B., "Symbolic Determination of Redundant Measurements for Plant Instrumentation Design", aceptado para su publicación en ENPROMER 2001, (3<sup>th</sup> Congress on Process Engineering for the MERCOSUR) a realizarse en Santa Fe (Argentina) del 16 al 20 de septiembre de 2001.

---

<sup>1</sup> Este trabajo fue presentado en la 4th World Multiconference on Circuits, Systems, Communications & Computers, (CSCC 2000), Vouliagmeni (Grecia). Organizado por WSES y IEEE. 10 al 15 de julio del 2000.

Ponzoni I., Vazquez G.E., Sánchez M.C., Brignole N.B., "A Computer-Aided DSS for Observability Analysis", 4<sup>th</sup> World Conference on Computers organizada por WSES y IEEE (dentro del marco de la 4th World Multiconference on Circuits, Systems, Communications & Computers, CSCC 2000), Vouliagmeni (Grecia). 10 al 15 de julio del 2000. Publicado en el CD del Congreso, ISBN: 960-8052-19X.

Ponzoni I., Vazquez G.E., Sánchez M.C., Brignole N.B., "A Parallel Algorithm for Observability Analysis on Networks of Workstations", AIChE Annual Meeting 1999, Dallas (Estados Unidos). 31 de octubre al 5 de noviembre de 1999. Publicado en el CD-ROM del congreso, paper 237i.

Ponzoni I., Sánchez M.C., Brignole N.B., "Análisis de observabilidad para problemas fuertemente no lineales", ENPROMER'99 (2<sup>nd</sup> Congress on Process Engineering for the MERCOSUR), Florianópolis (Brasil). 30 de agosto al 2 de septiembre de 1999, pp. 659-660. Trabajo completo publicado en el CD-ROM del congreso.

Vazquez G.E., Ponzoni I., Brignole N.B., "Parallel Depth-First Search on Clusters of Workstations", SIAM Annual Meeting 1999, Atlanta (Estados Unidos). 10 al 12 de mayo de 1999, pp. 198.

Vazquez G.E., Ponzoni I., Sánchez M.C., Brignole N.B., "ModGen: A Model Generator for Instrumentation Analysis. Industrial Application using New Observability Techniques", AIChE Annual Meeting 1998, Miami (Estados Unidos). 15 al 20 de noviembre de 1998. Publicado en el CD-ROM del congreso.

Ponzoni I., Sánchez M.C., Brignole N.B., "Constrained Partitioning of Sparse Matrices Using Graph Decompositions", SIAM Annual Meeting 1998, Toronto (Canadá). 13 al 17 de julio de 1998, pp. 87.

Ponzoni I., Sánchez M.C., Brignole N.B., "A New Hybrid Approach for Instrumentation Design of Chemical Plants", IV WCCM (4<sup>th</sup> World Congress on Computational Mechanics), Buenos Aires (Argentina). 29 de junio al 2 de julio de 1998. Publicado en el CD-ROM del congreso.

Ponzoni I., Sánchez M.C., Brignole N.B., "A Graph-Based Algorithm for Constrained Partitioning of Sparse Matrices", IV WCCM (4<sup>th</sup> World Congress on Computational Mechanics), Buenos Aires (Argentina). 29 de junio al 2 de julio de 1998. Publicado en el CD-ROM del congreso.

Ponzoni I., Sánchez M.C., Brignole N.B., "A New Partitioning Algorithm for Classification of Variables in Process Plant Monitoring", AIChE 1997 Annual Meeting, Los Angeles (Estados Unidos). 16 al 21 de noviembre de 1997. Publicado en el CD-ROM del congreso.

Ponzoni I., Sánchez M.C., Brignole N.B., "Estudio del Desempeño de Algoritmos de Clasificación de Variables No Medidas en Problemas de Instrumentación", ENPROMER'97 (1<sup>st</sup> Congress on Process Engineering for the MERCOSUR), Bahía Blanca (Argentina). 1 al 4 de septiembre de 1997, pp. 27-28.

Ponzoni I., Sánchez M.C., Brignole N.B., “CDHG: un nuevo Algoritmo de Particionamiento basado en la Detección de Ciclos en Hipergrafos”, ENPROMER’97 (1<sup>st</sup> Congress on Process Engineering for the MERCOSUR), Bahía Blanca (Argentina). 1 al 4 de septiembre de 1997, pp. 29-30.

En Congresos Nacionales con Comité de Revisión.

Ponzoni I., Brignole N.B., Bandoni A.J., “Estudio de Instrumentación para una Planta de producción de Amoniaco empleando un Nuevo Algoritmo de Clasificación”, AADECA’98 (Asociación Argentina de Control Automático), Buenos Aires (Argentina). 17 al 21 de agosto 1998, Vol. 1, pp. 59-64.

Ponzoni I., Sánchez M.C., Brignole N.B., “An Efficient Implementation of the First Least-Connected Node Algorithm”, X ENIEF (10<sup>o</sup> Congreso sobre Métodos Numéricos y sus Aplicaciones), San Carlos de Bariloche (Argentina). 10 al 14 de noviembre de 1997.

Ponzoni I., Sánchez M.C., Brignole N.B., “Permutación de matrices ralas a una FTB inferior específica mediante descomposición de grafos”, WAIT’97 (Primer Workshop Argentino de Informática Teórica), realizado en el marco de las XXVI JAIIO (Jornadas Argentinas de Informática e Investigación Operativa). Buenos Aires (Argentina). 11 de agosto de 1997, pp. 139-148.

Ponzoni I., Sánchez M.C., Brignole N.B., “Algoritmos eficientes en problemas de asignación”, X ENIEF (9<sup>o</sup> Congreso sobre Métodos Numéricos y sus Aplicaciones), San Carlos de Bariloche (Argentina). 6 al 10 de noviembre de 1995.

#### Reportes Internos

Ponzoni, I.; Vazquez, G.E.; M.C.; Brignole, N.B. A Parallel Depth-First Search on Heterogeneous Distributed Environments. Internal Report, PLAPIQUI (UNS-CONICET), Bahía Blanca, Argentina, mayo de 1999.

# *INTRODUCCIÓN*

Las ciencias de la computación constituyen una de las ramas del conocimiento de más fuerte crecimiento en los últimos 20 años. La computación como disciplina científica ha ido expandiéndose tan rápidamente hacia distintos y variados campos de aplicación que hoy en día resulta difícil encontrar un área de investigación científico-tecnológica que no incorpore conceptos y herramientas de esta disciplina. Asimismo, la conformación de grupos de investigación interdisciplinarios, integrados por profesionales de distintas especialidades, constituye la tendencia actual para desarrollar proyectos complejos pues es la forma más eficiente de encontrar soluciones de alta calidad para numerosos desafíos tecnológicos de este nuevo siglo.

En Ingeniería de Procesos el surgimiento de las computadoras permitió el tratamiento ágil y eficiente de problemas duros que requieren manejar y analizar grandes volúmenes de

información heterogénea. Los primeros paquetes integrales para simulación de plantas con cierta adaptabilidad a la configuración del proceso surgieron a fines de la década del sesenta con el advenimiento del CHESS (Motard *y col.*, 1968), que corría en computadoras IBM 360 ó SDS SIGMA 7. Posteriormente, apareció FLOWTRAN (Rosen y Pauls, 1977) que al principio pertenecía a la compañía Monsanto y luego fue adquirido como simulador base para el proyecto ASPEN (Advanced Simulator for Process Engineering), originalmente fundado por el Departamento de Energía de EE.UU. en 1976. Este equipo produjo una versión muy mejorada del producto en 1981. En la misma época surgió PROCESS® (1979) y ambos paquetes continuaron evolucionando entre los líderes hasta nuestros días. En el área de monitoreo de plantas, los primeros paquetes integrales surgieron recién a fines de la década del ochenta sobre la base de los desarrollos existentes en simulación y optimización, siendo perfeccionados gradualmente hasta la actualidad. En esta línea, los desarrollos comerciales de software de amplio espectro más conocidos son DATACONTM; VALI® y SIGMAFINE®. Estos paquetes efectúan reconciliación de datos, utilizando métodos numéricos para analizar las mediciones provenientes de la planta con diversos objetivos incluyendo estudios de rentabilidad, calibración de sensores, detección de fallas y optimización del control.

Una revisión completa de las herramientas computacionales para simulación, tanto comerciales como académicas, desarrolladas hasta fines de la década del ochenta puede consultarse en Biegler (1989). Los paquetes de software de este período están codificados en FORTRAN, que era el lenguaje de programación más ampliamente usado. Para ser útiles en la práctica, los programas permitían que el usuario integrara en forma directa rutinas externas también escritas en FORTRAN. En vista del gran tamaño y complejidad de muchos problemas, inmediatamente surgió la necesidad de usar estructuras de datos dinámicas para optimizar el uso de memoria. En tal sentido, el lenguaje FORTRAN tenía severas limitaciones, algunas de las

cuales fueron superadas en esa época con el desarrollo del prototipo PLEXSYS (Evans y col., 1977). Con respecto a la evolución del software para simulación en la década de los noventa, Biegler y col. (1997) presentan un apéndice con una lista completa de sistemas de modelado, simuladores de procesos, bancos de datos, herramientas de síntesis y manejo de información. Un hito importante de esta época es el vuelco hacia el uso masivo de las computadoras personales, lo cual obligó a la actualización de los paquetes de software existentes para su empleo en estas arquitecturas. Esta revolución conlleva una diversificación de los lenguajes de programación y la consiguiente necesidad de proveer entornos flexibles con capacidad para integrar rutinas codificadas en diversos lenguajes.

En cuanto a los recursos humanos, el desarrollo de los primeros programas para este tipo de aplicaciones estaba principalmente en manos de ingenieros químicos, quienes poseían un profundo conocimiento del problema y nociones básicas de programación, lo cual resultaba suficiente para simulación de equipos individuales y sectores de planta pequeños en el ámbito académico. A medida que las metas se volvieron más ambiciosas, volcándose hacia paquetes globales con múltiples facilidades y usuarios del ambiente industrial, las compañías líderes comenzaron a asignar las tareas de desarrollo de este tipo de programas a grandes equipos de trabajo conformados por ingenieros químicos, algunos provenientes de la academia y otros con experiencia industrial, trabajando con el soporte de especialistas en computación. En los últimos quince años la acelerada evolución tanto del hardware como del software, junto con la aparición de nuevas tecnologías, fueron tornando cada vez más compleja la tarea de desarrollar programas poderosos, eficientes y confiables que resulten atractivos y competitivos dentro de los estándares de calidad actuales. Por tal motivo, la labor conjunta de especialistas en Ciencias de la Computación e Ingenieros Químicos resulta ahora imprescindible para la obtención de productos de buena calidad.

## 1.1. OBJETIVOS Y ALCANCE DE LA TESIS

Un problema de especial interés dentro del área de monitoreo de procesos es el *análisis de observabilidad*. Este estudio se realiza a partir de una representación de “estado estacionario” de la planta, es decir una combinación de condiciones tal que nada cambia en el tiempo. Así, el funcionamiento en estado estacionario de una planta industrial puede ser modelado matemáticamente a través de un sistema de ecuaciones algebraicas no lineales. En la práctica, algunas de las variables del modelo tienen un valor conocido a través de mediciones efectuadas sobre la planta, mientras que las restantes constituyen el conjunto de variables no medidas del proceso. El análisis de observabilidad busca determinar cuáles variables no medidas pueden ser calculadas a partir de las mediciones y las ecuaciones del modelo.

Esto permite aprovechar el modelo matemático del proceso para reducir la cantidad de mediciones en el diseño y corroborar las lecturas de los instrumentos durante la operación. Un análisis de observabilidad correcto y riguroso permite obtener una mejor configuración de la instrumentación del proceso industrial lo cual redundará en múltiples beneficios. Contar con un conocimiento completo y confiable del funcionamiento de la planta permite lograr ganancias en términos de calidad, medio ambiente y normas de seguridad, ahorrando simultáneamente dinero y recursos.

Las metodologías para clasificación de variables no medidas existentes en la literatura poseen grandes deficiencias, tanto en robustez como en rango de aplicación, cuando se desea usarlas para sistemas de ecuaciones no lineales (ver capítulo 3). Por tal motivo, el empleo de estas técnicas en plantas con modelos matemáticos complejos no resulta confiable. Este hecho constituye nuestra principal motivación para desarrollar este trabajo.

El objetivo central de esta tesis es el desarrollo de nuevos algoritmos de observabilidad que resulten eficientes y robustos en el tratamiento de plantas industriales grandes y complejas, en donde los modelos matemáticos requieren de ecuaciones fuertemente no lineales para brindar una representación realista del problema. En particular, se desea obtener técnicas con un amplio rango de aplicabilidad.

La metodología de clasificación de variables más apropiada para alcanzar estas metas se basa en efectuar reordenamientos estructurales de la matriz de ocurrencia asociada al sistema de ecuaciones que modela la planta. En tal sentido, la teoría de grafos constituye una herramienta ideal para realizar estas permutaciones matriciales en forma eficiente y precisa. Durante el transcurso de esta tesis se estudió en profundidad la conexión existente entre descomposición de grafos y particionamiento de matrices a fin de proponer nuevas alternativas para llevar adelante un análisis de observabilidad estructural basado en particionamiento de grafos.

El objetivo final de esta tesis es el desarrollo de un sistema de soporte de decisión para análisis de observabilidad en plantas industriales, que asista al ingeniero de procesos en la ardua tarea de tomar decisiones respecto a la configuración de la instrumentación. La meta es diseñar e implementar un paquete de software que englobe toda la experiencia y conocimiento obtenidos a través de estas investigaciones. En tal sentido, además de un método eficiente y robusto para clasificación de variables, el programa debe brindar herramientas de software que ayuden al usuario en el análisis e interpretación de los resultados obtenidos por el algoritmo de observabilidad a fin de poder tomar las mejores decisiones respecto a la remoción o incorporación de sensores a la planta.

## **1.2. CONTENIDOS Y ESTRUCTURA DE LA TESIS**

La tesis está organizada en ocho capítulos. En el primero, se introdujo brevemente la problemática y se delinearon los principales objetivos de este trabajo. En el capítulo 2 se presentan todos los conceptos básicos de teoría de grafos requeridos para la comprensión de los algoritmos propuestos. El capítulo 3 introduce al lector en el problema de clasificación de variables en plantas de procesos, brindando un estudio crítico de las principales virtudes y deficiencias de las principales metodologías de observabilidad existentes en la actualidad.

En los capítulos 4, 5 y 6 se presentan los distintos algoritmos propuestos por el tesista para efectuar la clasificación de variables empleando distintas clases de grafos. En el capítulo 7 se discute el diseño e implementación de un sistema de soporte de decisión para análisis de observabilidad. Finalmente, en el capítulo 8, se exponen las principales conclusiones de este trabajo y se enuncian lineamientos y sugerencias para futuras investigaciones, mejoras y nuevas aplicaciones.

# *CONCEPTOS BÁSICOS DE TEORÍA DE GRAFOS*

## **2.1. INTRODUCCION**

Los grafos constituyen una de las herramientas de modelado más ampliamente utilizadas en un sinnúmero de aplicaciones en la actualidad. Sus orígenes se remontan al “*problema de los puentes sobre el río Pregel de Königsberg*”, publicado por Leonhard Euler en 1736. A posteriori, durante el siglo XIX, importantes resultados fueron obtenidos dentro de esta disciplina, y a principios de la tercera década del siglo XX surgieron los primeros intentos por establecer un marco conceptual riguroso para esta nueva teoría matemática.

No obstante estos avances, es recién a mediados del siglo XX que el modelado a través de grafos logra una fuerte expansión gracias al advenimiento de las computadoras (Bellman y *co.*, 1970; Christofides 1975; Even, 1979; Golumbic, 1980). Básicamente, un grafo es una abstracción útil de muchas situaciones del mundo real, sobre todo cuando el problema está relacionado esencialmente con la estructura de interconexión existente entre distintos objetos. Por tal motivo, los grafos resultan una estructura matemática ideal para modelar un amplio espectro de aplicaciones. En tal sentido, una “*estructura edilicia*” en Ingeniería Civil, un “*circuito digital*” en Ingeniería Electrónica, un “*sociograma*” o un “*esquema de organización*” en Sociología o Economía, una “*red de computadoras*” en Informática ó una “*estructura molecular*” en Química, por citar sólo algunas aplicaciones, son ejemplos claros de problemas que pueden modelarse en forma natural mediante el uso de grafos.

Por otra parte, los grafos poseen una estrecha relación con la teoría de matrices ralas y en particular con los métodos de reordenamiento y manipulación estructural de dichas matrices (Bunch y Rose, 1976; Pissanetzky, 1984; Duff y *co.*, 1997). Dado que esta tesis se centra en el desarrollo de algoritmos para el reordenamiento estructural de matrices de ocurrencia ralas asociadas a los sistemas de ecuaciones que modelan procesos industriales, la teoría de grafos constituye una base rigurosa a partir de la cual fundamentar las técnicas y metodologías propuestas en este trabajo.

En este capítulo se abordarán las nociones principales de teoría de grafos aplicadas a lo largo de este trabajo de tesis. La mayor parte de los conceptos, teoremas y demostraciones fueron extraídos del libro de Gibbons (1994). En primer lugar se presentarán definiciones elementales relativas a grafos no dirigidos y digrafos para luego introducir la noción de búsqueda en profundidad sobre grafos (DFS). Esta técnica constituye la base para varios de los métodos de análisis de observabilidad propuestos en esta tesis. En particular, se explicará en

detalle un algoritmo para la detección de las componentes fuertes de un grafo. Posteriormente, se estudiarán los conceptos de grafos bipartitos y algoritmos para la detección de pareamientos maximales. Luego, se establecerá la relación existente entre la permutación de matrices ralas y los conceptos de grafos. Finalmente, se presentarán algunas nociones básicas sobre hipergrafos.

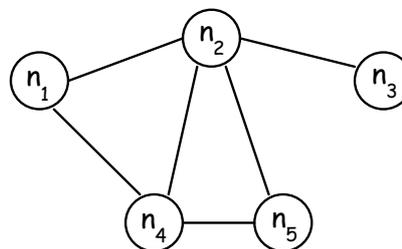
## 2.2. GRAFOS, DIGRAFOS Y NOCIONES DE CONECTIVIDAD

Un *grafo no dirigido* (o simplemente *grafo*)  $G$  consiste de un conjunto  $\mathbf{N}$  de *nodos* y una colección  $\mathbf{A}$  (no necesariamente un conjunto) de pares no ordenados de nodos, denominados *aristas*. Así, un grafo se representa simbólicamente como  $G = (\mathbf{N}, \mathbf{A})$ . En esta tesis sólo consideraremos grafos con  $\mathbf{N}$  y  $\mathbf{A}$  finitos. Luego, el *orden* de un grafo queda establecido por su número de nodos,  $|\mathbf{N}|$ , mientras que el *tamaño* de un grafo está dado por su cantidad de aristas,  $|\mathbf{A}|$ . Si  $u$  y  $v$  pertenecen a  $\mathbf{N}$  y existe un par no ordenado  $a = (u, v)$  en  $\mathbf{A}$ , se dice que  $a$  une  $u$  y  $v$ , ó que existe una arista  $a$  entre  $u$  y  $v$ . En este caso, también se dice que  $u$  y  $v$  son *incidentes* en  $a$ , ó que  $a$  tiene *incidencia* en  $u$  y  $v$ . Por otra parte, diremos que  $u$  es *adyacente* a  $v$  y  $v$  es *adyacente* a  $u$ . Finalmente, la forma gráfica de representar un grafo es empleando círculos o puntos para especificar los nodos y líneas para trazar las aristas. La figura 2.1 muestra un ejemplo de esta notación.

$G(\mathbf{N}, \mathbf{A})$  con:

$$\mathbf{N} = \{n_1, n_2, n_3, n_4, n_5\}$$

$$\mathbf{A} = \{(n_1, n_2), (n_1, n_4), (n_2, n_3), \\ (n_2, n_4), (n_2, n_5), (n_4, n_5)\}$$



**Figura 2.1.** Representación gráfica de un grafo no dirigido  $G$ .

Cuando dos o más aristas unen un mismo par de nodos, estas se denominan **aristas paralelas**, mientras que una arista cuyos extremos coinciden se denomina **bucle**. Se dice que  $G$  es un **grafo simple**, si  $G$  no contiene aristas paralelas ni bucles. Además, un grafo simple  $T$  se dice un **árbol** si dados dos nodos  $u$  y  $v$  de  $T$ , existe un único camino entre  $u$  y  $v$ . Por otra parte, un grafo  $S = (\mathbf{N}_S, \mathbf{A}_S)$  es un **subgrafo** de  $G = (\mathbf{N}, \mathbf{A})$  si  $\mathbf{N}_S$  es un subconjunto de  $\mathbf{N}$  y  $\mathbf{A}_S$  es un subconjunto de  $\mathbf{A}$ . Se dice que un subgrafo  $G'$  de  $G$  es un **subgrafo inducido** por el subconjunto de nodos  $\mathbf{N}'$ ,  $\mathbf{N}' \subseteq \mathbf{N}$ , si el grafo  $G'$  está conformado por todas las aristas de  $G$  con ambos extremos incluidos en  $\mathbf{N}'$ .

Los grafos no dirigidos resultan apropiados para modelar situaciones donde las relaciones existentes entre los distintos elementos del modelo son bidireccionales. Por ejemplo, el clásico problema del viajante, el cual consiste en determinar el camino más corto que pasa por un conjunto de ciudades, se puede resolver usando grafos no dirigidos. Esto se debe a que las rutas (representadas con las aristas del grafo) conectan ciudades (nodos) en ambos sentidos. Sin embargo, existen problemas en donde estos vínculos son unidireccionales. En muchos de estos casos, es necesario emplear grafos que tengan en cuenta la dirección (digrafos).

Un grafo  $G = (\mathbf{N}, \mathbf{A})$  se denomina **grafo dirigido** o **digrafo** si  $\mathbf{N}$  es un conjunto finito de nodos y  $\mathbf{A}$  un conjunto finito de pares ordenados. Si bien algunos autores se refieren a los pares ordenados como **arcos**, nosotros utilizaremos el término **aristas** tanto para los grafos no dirigidos como para los digrafos. Luego, una arista  $e = (u, v)$  de un digrafo  $G$  es una arista dirigida desde  $u$  hacia  $v$ . También se dice que la arista  $e$  parte de  $u$  y llega a  $v$ , ó que  $v$  es adyacente a  $u$ . Finalmente, la representación gráfica de un digrafo es similar a la empleada para los grafos no dirigidos sólo que ahora las aristas son flechas cuyas cabezas indican la dirección de las mismas. El **grafo subyacente** a un digrafo es el grafo no dirigido que se obtiene al

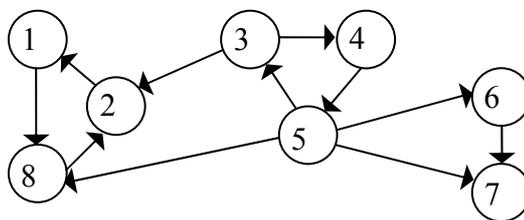
remover las direcciones de las aristas. La figura 2.2. muestra un digrafo extraído del libro de Gibbons (1994).

Sean  $u$  y  $v$  dos nodos de un grafo  $G$ . Una **caminata** (*walk*) entre  $u$  y  $v$  en el grafo es una secuencia finita y alternada  $u = n_0, a_1, n_1, a_2, n_2, a_3, \dots, a_k, n_k = v$  de nodos y aristas de  $G$  tales que la arista  $a_i$  en la secuencia une los nodos  $n_{i-1}$  y  $n_i$ . Los nodos y aristas en una caminata no son necesariamente distintos. La **longitud** de una caminata está dada por su número de aristas. Una caminata que no posee aristas repetidas se conoce como un **rastro** (*trail*). Asimismo, un **camino** o **sendero** (*path*) es una caminata que no contiene nodos repetidos. Obviamente, un camino es un caso particular de un rastro. Una **caminata cerrada** (*closed walk*) en un grafo es una caminata que empieza y termina en el mismo nodo. A su vez, una caminata cerrada que no contiene nodos repetidos se conoce como un **circuito** (*circuit*); si, además, tampoco tiene nodos repetidos, se denomina un **ciclo** (*cycle*).

$G = (\mathbf{N}, \mathbf{A})$  con:

$\mathbf{N} = \{1, 2, 3, 4, 5, 6, 7, 8\}$

$\mathbf{A} = \{(1,8), (2,1), (3,2), (3,4), (4,5), (5,3), (5,6), (5,7), (5,8), (6,7), (8,2)\}$



**Figura 2.2.** Representación gráfica de un digrafo  $G$ .

Dos nodos  $u$  y  $v$  se dicen conectados si existe un camino entre  $u$  y  $v$ . La conexión es una relación de equivalencia sobre el conjunto de nodos de un grafo que particiona a  $\mathbf{N}$  en los subconjuntos  $\mathbf{N}_1, \mathbf{N}_2, \dots, \mathbf{N}_k$ , donde dos nodos están conectados si y solo si pertenecen a la misma partición. Los subgrafos inducidos de  $\mathbf{N}_1, \mathbf{N}_2, \dots, \mathbf{N}_k$  se conocen como las **componentes conexas** del grafo. Se denomina **grafo conectado** a cualquier grafo que posea una única componente conexa. Un **subgrafo de expansión**  $S$  de un grafo conectado  $G = (\mathbf{N}, \mathbf{A})$  es un

subgrafo conectado de  $G$  que contiene todos los nodos de  $\mathbf{N}$ . Si en particular  $S$  es un árbol, se dice que  $S$  es un *árbol de expansión* de  $G$ .

En cuanto a los digrafos, la mayoría de los conceptos anteriores se extienden de manera natural. Una *caminata dirigida* es una secuencia finita  $u = n_0, a_1, n_1, a_2, n_2, a_3, \dots, a_k, n_k = v$  de nodos y aristas de  $G$ , donde la arista  $a_i$  en la secuencia parte del nodo  $n_{i-1}$  y llega al nodo  $n_i$ . Luego, un *rastro dirigido* es una caminata que no posee nodos repetidos y un *camino dirigido* es una caminata sin aristas repetidas. Análogamente, los términos *circuito dirigido* y *ciclo dirigido* refieren a los conceptos de circuitos y ciclos en el contexto de digrafos.

Finalmente, un digrafo  $G = (\mathbf{N}, \mathbf{A})$  está *fuertemente conectado* si existe un camino desde  $u$  a  $v$  y un camino desde  $v$  a  $u$  para cada par distinto de vértices  $u$  y  $v$ , con  $u, v \in \mathbf{N}$ . Si un digrafo no está fuertemente conectado entonces puede ser partido en subgrafos fuertemente conectados; dichos subgrafos se denominan componentes fuertemente conectadas, ó simplemente *componentes fuertes*, de  $G$ . Cuando un digrafo  $G$  no está fuertemente conectado pero su grafo subyacente está conectado, se dice que  $G$  está *débilmente conectado*.

### 2.3. BÚSQUEDAS EN PROFUNDIDAD SOBRE GRAFOS

La mayoría de los algoritmos de grafos requieren de un método sistemático para visitar sus nodos. En particular, la *búsqueda en profundidad*, conocida en inglés como *Depth-First Search (DFS)*, es una técnica ampliamente usada que posee características que contribuyen a obtener algoritmos muy eficientes.

### 2.3.1. ALGORITMOS BÁSICOS

Consideremos en primera instancia la DFS sobre grafos no dirigidos únicamente. Supongamos que el procedimiento está actualmente en el nodo  $v$ . Luego, el paso general en la búsqueda requiere visitar ahora el próximo nodo adyacente a  $v$  que aún no haya sido alcanzado. Si no existe un nodo en tales condiciones, la búsqueda retorna al nodo  $u$ , el cual había sido visitado justo antes que  $v$ . El paso general de la búsqueda se repite hasta que todos los nodos, de la componente del grafo que se está explorando, hayan sido visitados.

Las aristas recorridas durante una DFS conforman un árbol de expansión para cada componente del grafo explorado. El conjunto de árboles generados se denomina *expansión forestal en profundidad F* de  $G$ . Esto establece una partición del conjunto de aristas  $\mathbf{A}$  de  $G$  en los subconjuntos  $\mathbf{A}_F$ , y  $\mathbf{R} = \mathbf{A} - \mathbf{A}_F$ . Las aristas en  $\mathbf{R}$  se denominan aristas de retroceso debido a que conectan nodos con sus ancestros. Los algoritmos 2.1 y 2.2 presentan la secuencia de pasos para efectuar la DFS y la DFS transversal sobre grafos.

#### Algoritmo 2.1. DFS sobre grafos (DFS).

```

Datos de entrada:  $i, v, \mathbf{N}, \mathbf{A}$ 
Datos de entrada-salida:  $\mathbf{A}_F$ 

DFI( $v$ )  $\leftarrow i$ .
 $i \leftarrow i+1$ .
Para cada nodo  $v'$  adyacente a  $v$  hacer:
    Si DFI( $v'$ )=0
        entonces
             $\mathbf{A}_F \leftarrow \mathbf{A}_F \cup \{(v, v')\}$ .
            DFS( $i, v', \mathbf{N}, \mathbf{A}, \mathbf{A}_F$ ).
        fin-si
fin-para

```

#### Algoritmo 2.2. DFS transversal sobre grafos (DFST).

---

Datos de entrada:  $\mathbf{N}, \mathbf{A}$   
 Datos de salida:  $\mathbf{A}_F$

$i \leftarrow 1$ .  
 $\mathbf{A}_F \leftarrow \emptyset$ .  
**Para** todo  $v \in \mathbf{N}$  **hacer**:  $\text{DFI}(v) \leftarrow 0$ .  
**Mientras** exista algún  $u \in \mathbf{N}$  tal que  $\text{DFI}(u) = 0$  **hacer**  
      $\text{DFS}(i, u, \mathbf{N}, \mathbf{A}, \mathbf{A}_F)$ .

---

$\mathbf{A}_F$  guarda el conjunto de aristas visitadas y  $\text{DFI}(v)$  indica el orden en que fue visitado cada nodo durante la DFS. Cuando un nodo no ha sido visitado, el valor de DFI correspondiente permanece en cero.

Para el caso de una DFS sobre digrafos el algoritmo no cambia. El conjunto de árboles generados constituye ahora una *expansión forestal saliente en profundidad F* de  $G$ . La búsqueda se denomina saliente porque cada una de las aristas que conforman la forestación tienen sus cabezas apuntando en la dirección de las hojas de los árboles. Para el caso de digrafos, el conjunto de aristas  $\mathbf{A}$  queda partido en los siguientes cuatro tipos:

- $\mathbf{A}_F$  (aristas salientes) = {aristas que corresponden a la expansión forestal saliente en profundidad},
- $\mathbf{R}_1$  (aristas de retroceso) = {aristas que van de un nodo cualquiera a uno de sus ancestros},
- $\mathbf{R}_2$  (aristas de avance) = {aristas que no están en  $\mathbf{F}$  y que van de un nodo cualquiera a uno de sus descendientes},
- $\mathbf{C}$  (aristas de cruce) = {aristas que conectan un par de nodos correspondientes a distintos árboles de la forestación}.

Establecida esta clasificación, es posible enunciar y demostrar el siguiente teorema:

**Teorema 2.1.** Durante una DFS efectuada sobre un digrafo, si  $(u, v) \in \mathbf{C}$ , entonces se verifica que  $\text{DFI}(u) > \text{DFI}(v)$ .

*Demostración (por el absurdo).* Asumamos que vale lo contrario, es decir, que  $\text{DFI}(u) < \text{DFI}(v)$ . Esto implica que  $u$  fue visitado antes que  $v$ . Luego,  $v$  tiene que estar en la lista de nodos adyacentes a  $u$ . Analicemos entonces el tipo de la arista  $(u, v)$ . Si  $\text{DFI}(v)$  es asignado cuando se explora  $(u, v)$ , entonces  $(u, v)$  debe pertenecer a  $\mathbf{A}_F$ . La otra alternativa es que  $v$  haya sido visitado como un descendiente de  $u$ , pero no como un hijo directo, en cuyo caso  $(u, v)$  es una arista de avance ( $\mathbf{B}_2$ ). De este modo,  $(u, v)$  nunca puede pertenecer a  $\mathbf{C}$  y esto se contradice con la hipótesis del teorema. Luego, necesariamente debe verificarse que  $\text{DFI}(u) > \text{DFI}(v)$ .

### 2.3.2. DETECCIÓN DE LAS COMPONENTES FUERTES DE UN DIGRAFO USANDO DFS

La DFS sobre grafos resulta la base del algoritmo más eficiente empleado para la determinación de las componentes fuertes de un digrafo (Tarjan, 1972). Como se verá más adelante, esta técnica resulta de gran utilidad en la descomposición y reordenamiento de matrices. El método depende crucialmente del siguiente teorema:

**Teorema 2.2.** Si  $G_i = (\mathbf{N}_i, \mathbf{A}_i)$  es una componente fuerte de un digrafo  $G$  y sea  $F = (\mathbf{N}_F, \mathbf{A}_F)$  una expansión forestal saliente en profundidad, luego  $T_i = (\mathbf{N}_i, \mathbf{A}_i \cap \mathbf{A}_F)$  es un árbol.

*Demostración.* En primer término probaremos que cualquier par de nodos  $u, v \in \mathbf{N}_i$  tienen un ancestro en común. Podemos asumir sin pérdida de generalidad que  $\text{DFI}(u) < \text{DFI}(v)$ . Dado que  $u$  y  $v$  pertenecen a la misma componente fuerte, sabemos que existe un camino dirigido  $P$  entre  $u$  y  $v$ . Luego, sea  $w$  un nodo de  $P$  tal que  $\text{DFI}(w) < \text{DFI}(x)$  para todo  $x$  en  $P$ . Si trazamos el camino  $P$  desde  $u$ , en algún momento se alcanzará el nodo  $w$ .  $P$  puede pasar únicamente por nodos descendientes de  $w$ . Esto se debe a que todas las aristas que van desde nodos descendientes de  $w$  a nodos que no son descendientes de  $w$  pertenecen a  $\mathbf{C}$  (ver teorema 2.1) ó a

$\mathbf{R1}$  y poseen valores de DFI menores que  $w$ . Luego,  $w$  también es ancestro de  $v$ . Esto completa la primera parte de la demostración, pues se comprueba que existe un nodo  $w$  que es ancestro de todos los demás. Sea  $w$  la raíz del subárbol que contiene cada nodo de  $\mathbf{N}_i$ . Si  $x \in \mathbf{N}_i$  y si  $y$  es un nodo en el camino entre  $w$  y  $x$ . El resto de la demostración se completa estableciendo que  $y \in \mathbf{N}_i$ .

La raíz de  $T_i$  se denomina *raíz de la componente fuerte* de  $G_i$ , y se denota como  $r_i$ . El teorema anterior sugiere una manera natural de encontrar las componentes fuertes de un digrafo  $G$ . Es posible buscar las raíces  $r_1, r_2, \dots, r_k$ , en un orden tal que si  $i < j$  entonces, durante la DFS transversal efectuada sobre  $G$ ,  $r_i$  sea visitada por última vez antes de la última visita a  $r_j$ . Luego, por el teorema 1.2 y dado que  $r_j$  no puede ser descendiente de  $r_i$  si  $\text{DFI}(r_i) > \text{DFI}(r_j)$ , se puede deducir que  $G_i$  es un subgrafo inducido por todos los nodos de  $G$  que son descendientes de  $r_i$  pero que a su vez no son descendientes de  $r_1, r_2, \dots, r_{i-1}$ .

Para facilitar la identificación computacional de las raíces de las componentes fuertes definiremos un parámetro  $Q(v)$  como sigue:

$$Q(v) = \text{mínimo}(\{\text{DFI}(v)\} \cup \{\text{DFI}(v') \mid (x, v') \text{ está en } \mathbf{B}_1 \text{ o } \mathbf{C}\})$$

Donde  $x$  es un descendiente de  $v$  y la raíz  $r$  de la componente fuerte que contiene a  $v'$  es un ancestro de  $v$ .

El siguiente teorema constituye la base para un algoritmo de detección de las raíces de las componentes fuertes de un digrafo.

**Teorema 2.3.** En un digrafo  $G$ ,  $v$  es la raíz de una componente fuerte sí y solo sí  $Q(v) = \text{DFI}(v)$ .

*Demostración.* Notemos que, por definición,  $Q(v) \leq \text{DFI}(v)$ . En primer término demostraremos que si  $v$  es la raíz de una componente fuerte, luego  $Q(v) = \text{DFI}(v)$ . Supongamos, por el contrario, que  $Q(v) < \text{DFI}(v)$ . Luego, por la definición de  $Q(v)$ , existe un nodo  $v'$  tal que  $\text{DFI}(v') < \text{DFI}(v)$ .

Por otra parte,  $DFI(r) < DFI(v')$  y por lo tanto  $DFI(r) < DFI(v)$ . Pero  $r$  y  $v$  deben pertenecer a la misma componente fuerte, dado que hay un camino desde  $r$  a  $v$  y otro desde  $v$  a  $r$  vía  $(x, v')$ . Luego, dado que  $DFI(r) < DFI(v)$ ,  $v$  no puede ser la raíz de una componente fuerte. Esto contradice la hipótesis y por lo tanto debemos concluir que  $Q(v) = DFI(v)$ .

Ahora sólo nos resta demostrar que si  $v$  no es la raíz de una componente fuerte entonces  $Q(v) < DFI(v)$ . Esto se comprueba por el absurdo asumiendo que  $Q(v) = DFI(v)$ . Por definición de  $Q(v)$  sabemos que no existe un nodo  $v'$  tal que  $DFI(v') < DFI(v)$ . Dado que  $v$  no es raíz, algún otro nodo  $r$  debe ser la raíz. Luego, debe existir un camino  $p$  desde  $v$  a  $r$  que contiene al menos un nodo (tal vez  $r$ ) que no es descendiente de  $v$ . Asumamos que dicho nodo es  $v'$ . Es claro que  $r$  y  $v'$  pertenecen a la misma componente fuerte. La arista de  $p$  incidente en  $v'$  está en  $\mathbf{B}_1$  o  $\mathbf{C}$ . Luego  $DFI(v') < DFI(v)$ , lo cual es una contradicción. Por lo tanto  $Q(v) < DFI(v)$ .

Establecida la validez del teorema anterior, se puede definir un algoritmo para determinar las componentes fuertes de un digrafo, a partir del DFS transversal. Para esto resulta necesario establecer una manera recursiva de evaluar el  $Q(v)$  a fin de poder embeberlo dentro del DFS. Esto puede realizarse de la siguiente forma:

$$Q(v) = \text{mínimo}(\{DFI(v)\} \cup \{Q(v') \mid v' \text{ es un hijo de } v\} \cup \{DFI(v') \mid (v, v') \text{ está en } \mathbf{B}_1 \text{ o } \mathbf{C}, \\ \text{tal que la raíz de la componente fuerte que contiene a } v' \text{ es un ancestro de } v\})$$

Considerando este replanteo de  $Q$  se presentan los algoritmos 2.3 (**DFSCCD**) y 2.4 (**TCCD**). En la figura 2.3 se muestra el resultado de aplicar el algoritmo **TCCD** al digrafo presentado anteriormente en la figura 2.2. La figura 2.3 a) muestra la expansión forestal saliente en profundidad  $F$  de  $G$ , donde las aristas pertenecientes a  $\mathbf{A}_F$ , se indican con trazos continuos y las aristas de  $\mathbf{R}_1$ ,  $\mathbf{R}_2$  y  $\mathbf{C}$  con trazos discontinuos. La tabla de la derecha indica los valores

de DFI y Q asignados durante la búsqueda para cada nodo. La figura 2.3 b) muestra el digrafo con sus respectivas componentes fuertes agrupadas mediante círculos sombreados.

**Algoritmo 2.3.** DFS para detección de componentes fuertes de digrafos (**DFSCCD**).

*Datos de entrada:*  $i, v, \mathbf{N}, \mathbf{A}, P$   
*Datos de entrada-salida:*  $j, \mathbf{CC}$

DFI( $v$ )  $\leftarrow i$ .  
 Q( $v$ )  $\leftarrow$  DFI( $v$ ).  
 $i \leftarrow i+1$ .  
 Poner  $v$  en  $P$ .  
 Apilado( $v$ )  $\leftarrow$  verdadero.  
**Para** cada nodo  $v'$  adyacente a  $v$  **hacer:**  
   **Si** DFI( $v'$ )=0  
     **entonces**  
       DFSCCG( $i, v', \text{DFI}, \mathbf{N}, \mathbf{A}, P, j, \mathbf{CC}$ ).  
       Q( $v$ )  $\leftarrow$  min(Q( $v$ ), Q( $v'$ )).  
     **sino,**  
       **Si** DFI( $v'$ ) < DFI( $v$ ) **y** Apilado( $v'$ )  
         **entonces**  
           Q( $v$ )  $\leftarrow$  min(Q( $v$ ), DFI( $v'$ )).  
       **fin-si**  
     **fin-si**  
**fin-para**  
**Si** Q( $v$ )=DFI( $v$ )  
**entonces**  
   Desapilar de  $P$  todos los elementos hasta alcanzar  $v$ .  
   Guardar los elementos desapilados en  $\mathbf{CC}(j)$ , incluido  $v$ .  
   Desapilar  $v$ .  
   Setear Apilado( $u$ )  $\leftarrow$  falso, para todos los nodos en  $\mathbf{CC}(j)$ .  
    $j \leftarrow j+1$ .  
   {los elementos almacenados en  $\mathbf{CC}(j)$  constituyen la  
    $j$ -ésima componente fuerte del grafo con raíz  $v$ }  
**fin-si**

**Algoritmo 2.4.** Detección de todas las componentes fuertes de un digrafo (**TCCD**)

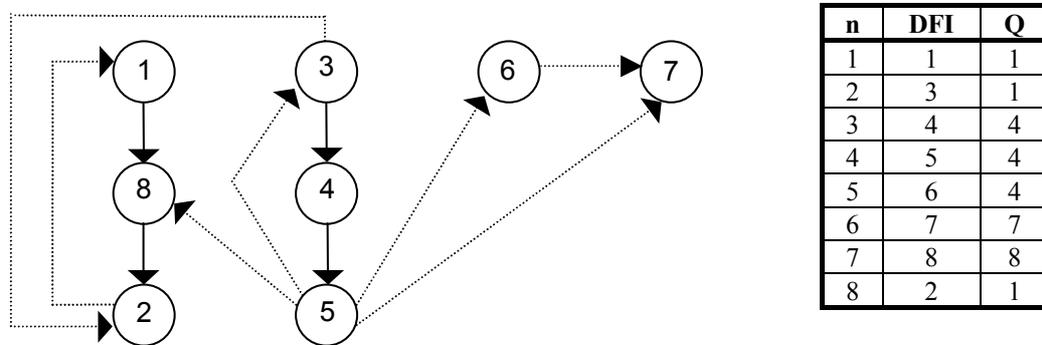
*Datos de entrada:*  $\mathbf{N}, \mathbf{A}$   
*Datos de entrada-salida:*  $\mathbf{CC}$

$i \leftarrow 1$ .  
 $j \leftarrow 1$ .  
 vaciar  $P$ .  
 vaciar  $\mathbf{CC}$ .

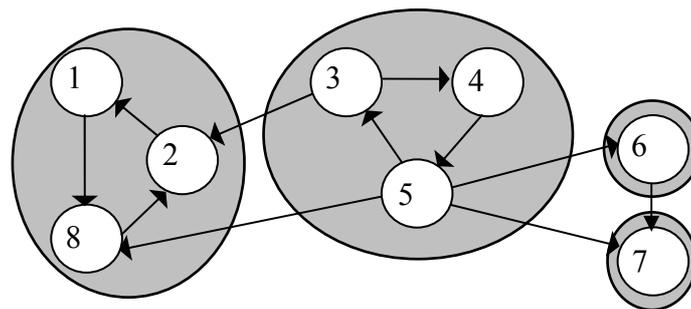
**Para** todo  $v \in \mathbf{N}$  **hacer:**  
   DFI( $v$ )  $\leftarrow 0$ .  
   Apilado( $v$ )  $\leftarrow$  falso.

```

fin-para
Mientras exista algún  $u$  tal que  $DFI(u)=0$  hacer:
    DFSCCG( $i, u, DFI, \mathbf{N}, \mathbf{A}, P, j, \mathbf{CC}$ ) .
fin-mientras
    
```



a) Expansión forestal de  $G$  y valores de DFI y  $Q$  de los nodos.



b) Componentes fuertes del grafo  $G$ .

Figura 2.3. Componentes fuertes detectadas por TCCD.

## 2.4. GRAFOS BIPARTITOS Y PAREAMIENTOS MAXIMALES

Un *grafo bipartito* (*bigrafo*) es un grafo simple en donde el conjunto de nodos puede ser particionado en dos subconjuntos disjuntos tales que todas las aristas del grafo posean un extremo en cada subconjunto. La representación matemática de un bigrafo está dada por la terna  $G = (\mathbf{N1}, \mathbf{N2}, \mathbf{A})$ , donde  $\mathbf{N1}$  y  $\mathbf{N2}$  son los conjuntos de nodos y  $\mathbf{A}$  el conjunto de aristas. En

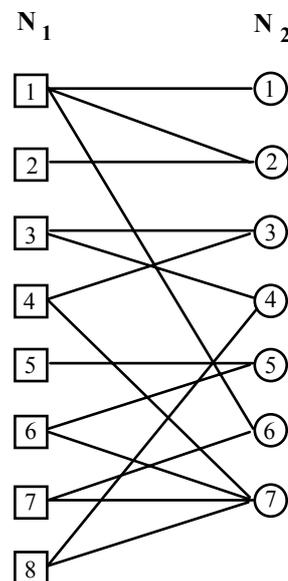
la figura 2.4 se muestra la representación gráfica de un bigrafo, la cual emplea cuadrados y círculos para esquematizar los nodos de  $\mathbf{N}_1$  y  $\mathbf{N}_2$  respectivamente.

$\mathbf{G} = (\mathbf{N}_1, \mathbf{N}_2, \mathbf{A})$  con:

$$\mathbf{N}_1 = \{nl_1, nl_2, nl_3, nl_4, nl_5, nl_6, nl_7, nl_8\}$$

$$\mathbf{N}_2 = \{n2_1, n2_2, n2_3, n2_4, n2_5, n2_6, n2_7\}$$

$$\mathbf{A} = \{(nl_1, n2_1), (nl_1, n2_2), (nl_1, n2_6), (nl_1, n2_7), (nl_3, n2_3), (nl_3, n2_4), (nl_4, n2_3), (nl_4, n2_7), (nl_5, n2_5), (nl_6, n2_5), (nl_6, n2_7), (nl_7, n2_6), (nl_7, n2_7), (nl_8, n2_4), (nl_8, n2_7)\}$$



**Figura 2.4.** Representación gráfica de un bigrafo  $\mathbf{G}$ .

Muchas de las aplicaciones modeladas a través de bigrafos requieren la búsqueda de pareamientos maximales (Asratian y *co.*, 1998; Karpinski y Rytter, 1998). Un **pareamiento**  $\mathbf{P}$  en un bigrafo  $\mathbf{G} = (\mathbf{N}_1, \mathbf{N}_2, \mathbf{A})$  es un subconjunto de  $\mathbf{A}$  tal que ninguna de las aristas de  $\mathbf{P}$  tienen extremos en común. La cardinalidad de un pareamiento está dado por el número de aristas que contiene. En tal sentido, se dice que  $\mathbf{P}$  es un **pareamiento maximal** para un bigrafo  $\mathbf{G}$  si no existe otro pareamiento  $\mathbf{P}'$  en  $\mathbf{G}$  tal que la cardinalidad de  $\mathbf{P}'$  sea mayor que la cardinalidad de  $\mathbf{P}$ . Nótese que un bigrafo puede tener más de un pareamiento maximal.

Antes de explicar cómo obtener un pareamiento maximal es necesario definir algunos conceptos auxiliares. Una **caminata alternante** sobre un bigrafo  $\mathbf{G}$  relativa a un pareamiento  $\mathbf{P}$  es una caminata cuyas aristas alternan en el pareamiento  $\mathbf{P}$ . Por ejemplo, si la primera arista del camino alternante no está en  $\mathbf{P}$ , luego las aristas en posiciones pares del camino pertenecen a  $\mathbf{P}$  y las impares no. Asimismo, un **sendero alternante** es una caminata alternante que no contiene

aristas repetidas y un *sendero aumentado* es un sendero alternante cuyos extremos corresponden a nodos no apareados por  $\mathbf{P}$ .

La *cardinalidad* de un pareamiento es el número de aristas que contiene. Un camino aumentado  $\mathbf{A}$  puede ser utilizado para incrementar la cardinalidad de un pareamiento  $\mathbf{P}$  removiendo de  $\mathbf{P}$  las aristas presentes en ambos conjuntos  $(\mathbf{A} \cap \mathbf{P})$  y agregando a  $\mathbf{P}$  las aristas que sólo están en  $\mathbf{A}$ . Es sencillo demostrar que cada camino aumentado  $\mathbf{A}$ , relativo a  $\mathbf{P}$ , permite aumentar en uno la cardinalidad del pareamiento. De esta manera, se alcanza un pareamiento maximal cuando no existen más caminos aumentados. En el algoritmo 2.5 se muestra como obtener un pareamiento maximal sobre bigrafos.

**Algoritmo 2.5.** Pareamiento maximal de un bigrafo (PMB).

```

Datos de entrada:  $\mathbf{R}, \mathbf{C}, \mathbf{A}$ 
Datos de salida:  $\mathbf{P}_M$ 

 $\mathbf{P}_M \leftarrow \emptyset.$ 
 $\mathbf{C}_U \leftarrow \emptyset.$ 
% Construcción de un pareamiento inicial
Para cada nodo  $c \in \mathbf{C}$  hacer:
  Aparear  $c$  con el primer nodo no apareado  $r$ , tal que  $r \in \mathbf{R}$ .
  Si no existe tal nodo  $r$  entonces  $\mathbf{C}_U \leftarrow \mathbf{C}_U \cup \{c\}$ 
fin-para

% Búsqueda de caminos aumentados
 $\mathbf{C}_{UN} \leftarrow \emptyset.$ 
Repetir
  Buscar un camino aumentado  $\mathbf{A}_u$  desde  $c$ , visitando únicamente
  nodos de  $\mathbf{R}$  que no han sido visitados antes durante este paso.
  Marcar todos los nodos alcanzados como visitados.
  Si se ha encontrado un camino aumentado  $\mathbf{A}$ 
    entonces
      Aumentar  $\mathbf{P}_M$  con  $\mathbf{A}_u$ .
    sino
      Incluir  $c$  en  $\mathbf{C}_{UN}$ .
  fin-si.
 $\mathbf{C}_U \leftarrow \mathbf{C}_{UN}$ .
 $\mathbf{C}_{UN} \leftarrow \emptyset.$ 
hasta: que no se encuentre ningún camino aumentado en un paso.

```

## 2.5. GRAFOS Y PERMUTACIÓN DE MATRICES

Existe una estrecha relación entre los grafos y las matrices. De hecho, una de las formas más utilizadas para representar grafos es justamente mediante matrices.

### 2.5.1. MATRICES DE ADYACENCIA

Sea  $G = (\mathbf{N}, \mathbf{A})$  un grafo donde  $\mathbf{N} = \{1, 2, \dots, n\}$ ; la **matriz de adyacencia del grafo  $G$**  se define como la matriz  $\mathbf{A} = [a_{ij}]$  de  $n \times n$ , donde los elementos  $a_{ij}$  fuera de la diagonal ( $i \neq j$ ) indican el número de aristas entre los nodos  $i$  y  $j$ , y los elementos  $a_{ii}$  sobre la diagonal expresan el doble del número de bucles en el nodo  $i$ . La figura 2.5 a) muestra la matriz de adyacencia correspondiente al grafo de la figura 2.1.

Es claro que la matriz de adyacencia de un grafo no dirigido es siempre simétrica. A su vez, la matriz de adyacencia de un grafo simple es binaria y todos sus elementos sobre la diagonal son iguales a cero. Asimismo, la **matriz de adyacencia del digrafo  $G$**  con  $n$  nodos es la matriz binaria  $\mathbf{A} = [a_{ij}]$  de  $n \times n$ , donde cada elemento  $a_{ij}$  con  $i \neq j$  será igual a 1 si existe una arista desde el nodo  $i$  hacia el nodo  $j$  y cero en el caso contrario, mientras que los elementos sobre la diagonal son iguales a cero. La matriz de adyacencia para un digrafo no es necesariamente simétrica. La figura 2.5 b) muestra la matriz de adyacencia del grafo de la figura 2.2.

$$\mathbf{M}(G) = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix} \qquad \mathbf{M}(G) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

a) Matriz de incidencia del grafo de la fig. 2.1    b) Matriz de incidencia del digrafo de la fig. 2.2

**Figura 2.5. Matrices de incidencia.**

**2.5.2. PERMUTACIÓN DE MATRICES CUADRADAS A FORMA TRIANGULAR EN BLOQUES**

La permutación de matrices ralas a formas especiales permite ahorrar tiempos de cómputo y espacio de memoria. En particular, el reordenamiento estructural a *forma triangular en bloques (FTB)* ha sido usado con éxito en la resolución de sistemas lineales, problemas de mínimos cuadrados, problemas de espacio nulo, particionamiento de matrices para procesamiento paralelo, entre otras aplicaciones (Pothen y Fan, 1990).

A fin de estudiar la permutación de matrices cuadradas resulta necesario introducir los siguientes conceptos. Una matriz cuadrada **A** se dice con *transversal llena* si su diagonal está libre de ceros. Para toda matriz cuadrada **A** estructuralmente no singular, es posible encontrar dos matrices de permutación **P** y **Q** tales que  $\mathbf{PAQ} = \mathbf{B}$ , donde **B** tiene transversal llena. Asimismo, si una matriz cuadrada **B** tiene transversal llena, es posible encontrar una matriz de permutación **P** tal que  $\mathbf{PAP} = \mathbf{C}$ , donde **C** tiene FTB. Cuando una matriz puede ser permutada a FTB se dice *reducible*. Si, en particular, una matriz **A** puede ser reordenada a FTB efectuando primero una permutación no simétrica para llevarla a transversal llena y realizando luego otra permutación simétrica para obtener la FTB, se dice que **A** es *bireducible*.

De lo anterior se desprende la necesidad de contar con dos métodos. En primer término, un algoritmo que dada una matriz cuadrada **A** estructuralmente no singular, obtenga la matriz **B** con su transversal llena. En segundo lugar, un algoritmo que dada **B** obtenga la matriz **C** correspondiente a su FTB.

La teoría de grafos puede ser empleada exitosamente a fin de resolver ambas tareas. Para obtener la máxima transversal de una matriz existen distintos métodos e implementaciones de los mismos (Hall, 1956; Hopcroft y Karp, 1973; Duff, 1981). La técnica más eficiente consiste

en asociar un bigrafo  $G = (\mathbf{R}, \mathbf{C}, \mathbf{A})$  a la matriz  $\mathbf{A}$  de forma tal que los nodos de  $\mathbf{R}$  y  $\mathbf{C}$  correspondan a las filas y columnas de  $\mathbf{A}$  respectivamente. De este modo, determinar la máxima transversal de  $\mathbf{A}$  es equivalente a hallar un pareamiento maximal  $\mathbf{P}$  para  $G$ , por lo que se puede aplicar directamente el algoritmo **PMB** descrito en la sección dedicada a bigrafos.

Una vez hallado el pareamiento maximal  $\mathbf{P}$ , la matriz  $\mathbf{A}$  puede ser reordenada de forma tal que si  $(r_i, c_j)$  es la  $k$ -ésima arista de  $\mathbf{P}$ , luego la  $i$ -ésima fila de  $\mathbf{A}$  será la  $k$ -ésima fila de  $\mathbf{B}$  y la  $j$ -ésima columna de  $\mathbf{A}$  será la  $k$ -ésima columna de  $\mathbf{B}$ . En otras palabras, los elementos sobre la diagonal de  $\mathbf{B}$  corresponderán a las aristas presentes en  $\mathbf{P}$ . Una vez obtenida  $\mathbf{B}$ , es posible encontrar su FTB utilizando digrafos.

Toda matriz cuadrada  $\mathbf{B}$  con transversal llena puede ser asociada a un digrafo  $G = (\mathbf{N}, \mathbf{A})$ . Si la matriz  $\mathbf{B}$  es de orden  $k$ , el digrafo tendrá  $k$  nodos  $(n_1, n_2, \dots, n_k)$  y la arista  $(n_i, n_j)$  pertenecerá a  $\mathbf{A}$  si y solo si  $\mathbf{B}_{ij} = 1$ . Cada elemento no nulo sobre la diagonal de  $\mathbf{A}$  corresponde a una autoarista  $(n_i, n_i)$ . En los casos en que  $\mathbf{B}$  tiene transversal llena, todas las autoaristas están presentes y usualmente no es necesario tenerlas en cuenta explícitamente.

Una de las propiedades de los digrafos es que son invariantes con respecto a las permutaciones simétricas. En otras palabras, cuando se efectúa una permutación simétrica sobre la matriz de adyacencia de un digrafo  $G$ , se obtiene la matriz de incidencia de un digrafo  $G'$ , el cual es equivalente a  $G$ , dado que sólo difiere en el nombre o numeración de los nodos. Es decir, el nodo  $n_i$  de  $G$  puede resultar ser ahora el nodo  $n_j$  de  $G'$ .

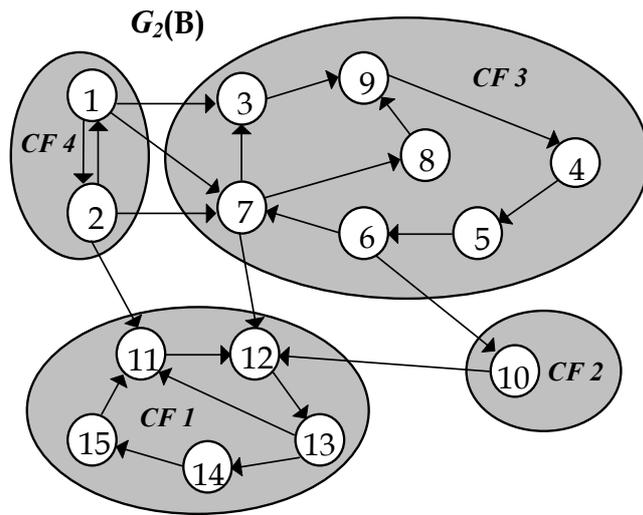
Por otra parte, las componentes fuertes del digrafo  $G$  se corresponden con los bloques cuadrados de la FTB de  $\mathbf{B}$ . Luego, mediante la aplicación del algoritmo **TCCD** es posible obtener la matriz  $\mathbf{C}$  reordenada a FTB mediante el reordenamiento de  $\mathbf{B}$  de forma tal que los elementos correspondientes a la primera componente fuerte de  $G$  aparezcan en las primeras filas

y columnas de **C**, seguidos de los elementos asociados a la segunda componente, y así hasta terminar.

En la figura 2.6 se presenta un ejemplo completo. En primer lugar, se muestra la matriz **A** y su bigrafo asociado  $G_1$ . Las aristas de  $G_1$ , indicadas con trazos más gruesos, corresponden al pareamiento maximal **P** encontrado con el algoritmo **PMB**. La matriz **B** es el resultado de permutar **A** siguiendo el orden de los nodos en el pareamiento **P**; la transversal de **B** es llena y sus elementos se muestran resaltados en negrita.  $G_2$  es el digrafo asociado a **B**, donde las componentes fuertes detectadas por **TCCD** están agrupadas mediante círculos sombreados. Finalmente se muestra la matriz **C** reordenada a FTB, donde los bloques correspondientes a las componentes fuertes están resaltados en gris.

$$\mathbf{A} = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \end{matrix} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \end{matrix}$$

$$\mathbf{B} = \begin{matrix} & \begin{matrix} 8 & 11 & 12 & 13 & 10 & 7 & 9 & 15 & 1 & 4 & 5 & 6 & 3 & 14 & 2 \end{matrix} & \begin{matrix} C/R \\ 8 \\ 6 \\ 13 \\ 1 \\ 3 \\ 5 \\ 15 \\ 11 \\ 9 \\ 7 \\ 4 \\ 14 \\ 12 \\ 10 \\ 2 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \end{matrix} & \begin{bmatrix} \mathbf{1} & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & \mathbf{1} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \mathbf{1} & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 1 \end{bmatrix} \end{matrix}$$



	11	12	13	14	15	10	3	4	5	6	7	8	9	1	2		
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12
1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	13
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	14
1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	15
0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	10
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	4
0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	5
0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	6
0	1	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	7
0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	8
0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	9
0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	1	0	1
1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	2

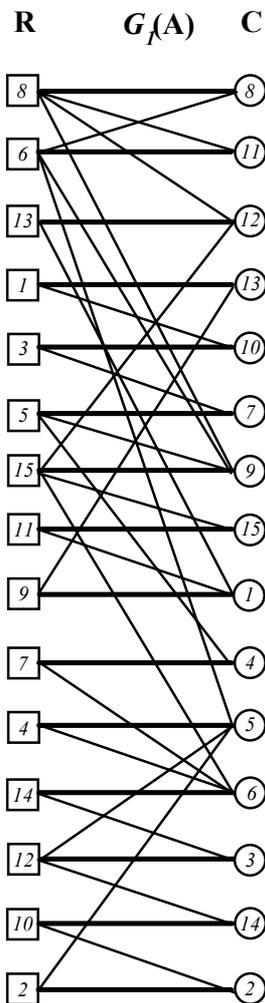


Figura 2.6. Obtención de la FTB aplicando los algoritmos PMB y TCCD.

Para terminar, cabe mencionar que existen varias implementaciones del algoritmo de Tarjan orientadas específicamente al reordenamiento de matrices cuadradas, entre las que se destacan las rutinas desarrolladas por Duff y Reid (1978) y Gustavson (1976).

### 2.5.3. PERMUTACIÓN DE MATRICES NO CUADRADAS A FORMA TRIANGULAR EN BLOQUES

Hasta ahora se ha analizado únicamente la permutación de matrices con transversal llena. Para el caso de matrices estructuralmente singulares los algoritmos existentes se basan en la descomposición de bigrafos propuesta por Dulmage y Mendelsohn (1958, 1963).

Básicamente, estos algoritmos asocian un bigrafo  $G = (R, C, A)$  a la matriz  $A$  y buscan un pareamiento maximal  $P$  empleando el algoritmo descrito en la subsección anterior. Una vez hallado  $P$  se procede a clasificar los nodos de  $R$  y  $C$  siguiendo la siguiente caracterización propuesta por Dulmage y Mendelsohn:

$VR = \{\text{nodos de } R \text{ alcanzables desde algún nodo de } R \text{ no apareado mediante un camino alternante}\}$

$HR = \{\text{nodos de } R \text{ alcanzables desde algún nodo de } C \text{ no apareado mediante un camino alternante}\}$

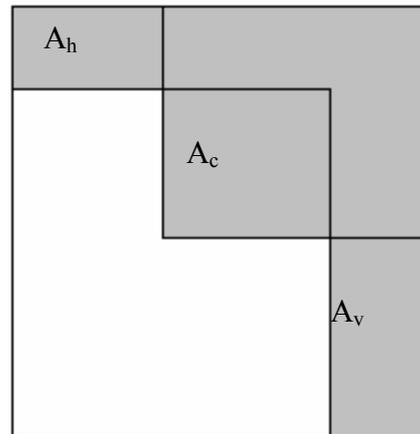
$SR = R \setminus (VR \cup HR)$

$VC = \{\text{nodos de } C \text{ alcanzables desde algún nodo de } R \text{ no apareado mediante un camino alternante}\}$

$HC = \{\text{nodos de } C \text{ alcanzables desde algún nodo de } C \text{ no apareado mediante un camino alternante}\}$

$SC = C \setminus (VC \cup HC)$

Esta clasificación permite descomponer la matriz asociada al bigrafo en bloques, los cuales reordenados en forma apropiada conducen a la FTB superior mostrada en la figura 2.7, donde sólo el bloque  $A_c$  es cuadrado y con transversal llena.



**Figura 2.7.** FTB superior obtenida usando la clasificación de Dulmage-Mendelsohn

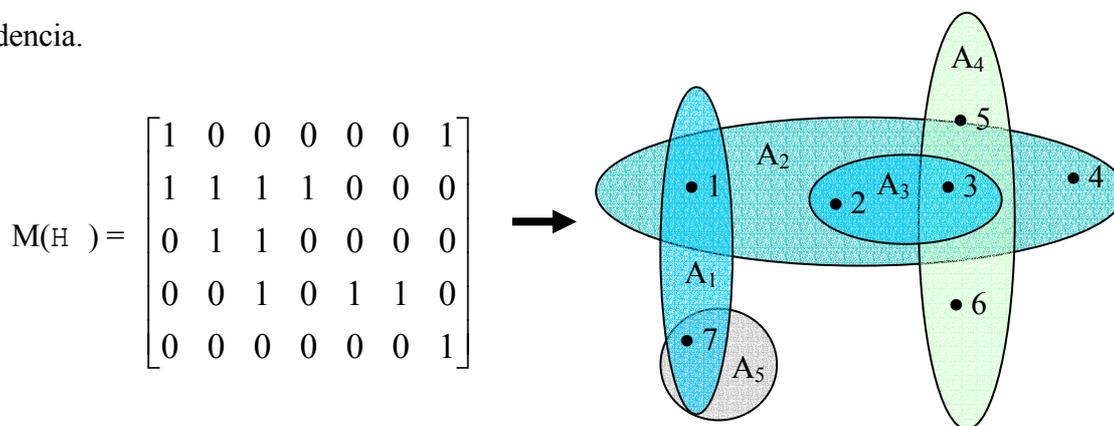
En 1990, Pothen y Fan presentaron una técnica de particionamiento para matrices no cuadradas la cual consiste de dos etapas. En la primera, denominada descomposición gruesa, se aplica la clasificación de Dulmage-Mendelsohn para obtener la FTB superior de la figura 2.7. En la segunda etapa, conocida como descomposición fina, se particionan las tres submatrices de la diagonal ( $A_h$ ,  $A_c$  y  $A_v$ ) en **bloques no reducibles**, es decir, en submatrices que no pueden ser descompuestas en más bloques.

Para las matrices  $A_h$  y  $A_v$  se emplea una DFS a fin de determinar todas sus componentes conexas. Cada una de estas componentes conformará un bloque no reducible de la FTB superior final. Por otra parte, el bloque  $A_c$  es particionado utilizando las técnicas presentadas para matrices cuadradas. De este modo, se obtiene una FTB superior donde todos los bloques son irreducibles.

## 2.6. HIPERGRAFOS

Para finalizar este capítulo, se presentan algunas nociones fundamentales sobre teoría de hipergrafos (Berge, 1987; Berge, 1988). Estos conceptos serán empleados por uno de los métodos propuestos en esta tesis, el cual será descrito en el capítulo 5.

Un **hipergrafo** es una familia  $\mathbf{H} = (\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_r)$  de subconjuntos  $\mathbf{A}_i$ , ninguno de ellos vacíos, y donde existe además un conjunto  $\mathbf{N} = \{n_1, n_2, \dots, n_c\}$ , tal que la unión de los  $\mathbf{A}_i$  es  $\mathbf{N}$  y no existe ningún par  $\mathbf{A}_i, \mathbf{A}_j$  en  $\mathbf{H}$  tal que  $\mathbf{A}_i = \mathbf{A}_j$  para  $i \neq j$ . Los elementos  $n_i$  y los subconjuntos  $\mathbf{A}_i$  se denominan nodos y aristas de  $\mathbf{H}$  respectivamente. Es interesante notar que cada arista es un conjunto de nodos. Esto es una diferencia substancial con respecto a los grafos clásicos, en donde una arista establece una relación entre uno o dos nodos únicamente. La **matriz de incidencia** de un hipergrafo  $\mathbf{H}$  es tal que sus filas corresponden a los nodos de  $\mathbf{N}$  y las columnas a las aristas de  $\mathbf{H}$ . La figura 2.8 muestra un hipergrafo y su correspondiente matriz de incidencia.



**Figura 2.8.** Un hipergrafo y su matriz de incidencia

El **orden** de un hipergrafo está dado por su número de aristas ( $\text{orden}(\mathbf{H}) = r$ ). Por otra parte, la **cardinalidad** de una arista  $\mathbf{A}$  es la cantidad de nodos contenidos en  $\mathbf{A}$ , y se indica como  $|\mathbf{A}|$ . Las aristas con cardinalidad uno se conocen como **autoaristas**. Por otra parte, un **ciclo** en un hipergrafo es una secuencia de la forma  $(n_1, \mathbf{A}_1, n_2, \mathbf{A}_2, \dots, n_k, \mathbf{A}_k, n_1)$ , para  $k \geq 2$ , donde  $n_1, n_2, \dots, n_k$  son diferentes nodos de  $\mathbf{N}$ ,  $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_k$  son diferentes aristas de  $\mathbf{H}$ , y  $\mathbf{A}_i$  contiene a  $x_i$  y  $x_{i+1} \forall i$ .

Un **hipergrafo parcial** de  $\mathbf{H}$  es un hipergrafo construido a partir de una familia de aristas  $\mathbf{H}'$  de  $\mathbf{H}$ , y su correspondiente conjunto de nodos  $\mathbf{N}'$  está formado por la unión de todas las

aristas de  $\mathbf{H}$ . Dado un hipergrafo  $\mathbf{H}$  con un conjunto  $\mathbf{N}$  de nodos, un *subgrafo inducido*  $\mathbf{H}_{\mathbf{N}_1}$  en  $\mathbf{H}$  es determinado por un subconjunto  $\mathbf{N}_1 \subseteq \mathbf{N}$ , donde  $\mathbf{H}_{\mathbf{N}_1} = (\mathbf{Z}_1, \mathbf{Z}_2, \dots, \mathbf{Z}_q)$  y  $\mathbf{Z}_k = \mathbf{A}_i \cap \mathbf{N}_1 \neq \emptyset$  para  $1 \leq k \leq q$  y  $1 \leq i \leq r$ . Nótese que  $q \leq r$ , la desigualdad estricta está contenida cuando  $\mathbf{A}_i \cap \mathbf{N}_1 = \emptyset$  para algún  $\mathbf{A}_i$ . Un *hipergrafo cuadrado* es cualquier hipergrafo cuyo orden es igual a su cantidad de nodos. Para este caso, la matriz de incidencia del hipergrafo será cuadrada. Finalmente, sea  $\mathbf{C} \subseteq \mathbf{N}$ ; luego, un *hipergrafo inducido cerrado*  $\mathbf{H}_{\mathbf{C}}$  de  $\mathbf{H}$  se define como el hipergrafo formado por todas las aristas de  $\mathbf{H}$  contenidas en  $\mathbf{C}$ , es decir,  $\mathbf{H}_{\mathbf{C}} = (\mathbf{A}_i : \mathbf{A}_i \subseteq \mathbf{C}, 1 \leq i \leq r)$ .

# ***CLASIFICACIÓN DE VARIABLES EN PROCESOS INDUSTRIALES***

## **3.1. INTRODUCCIÓN**

Dentro de la disciplina de Ingeniería de Procesos, el monitoreo es un área importante que se ocupa de tomar datos del proceso, registrarlos y analizarlos. Esto tiene gran interés desde el punto de vista económico ya que un buen monitoreo conduce a una operación eficiente y segura. El *análisis de instrumentación* consiste en definir la cantidad, tipo y ubicación de los sensores requeridos para lograr suficiente conocimiento del estado real de la planta en cualquier instante. Los estudios pueden efectuarse con el objeto de diseñar la instrumentación de una nueva planta o equipo individual, o bien con el propósito de modificar la instrumentación

existente (revamp). En cualquier caso, un análisis correcto y riguroso que permita lograr una instrumentación completa y confiable es fundamental para producir mejoras reales en la práctica, de modo de cumplir con los requerimientos de calidad, las reglamentaciones ambientales y las normas de seguridad, ahorrando al mismo tiempo dinero y recursos. En tal sentido, una instrumentación deficiente genera gastos por diversas razones. Por ejemplo, el desconocimiento del funcionamiento de algún sector puede provocar paradas de planta adicionales por carencia de un control regulatorio adecuado; además la presencia de instrumentos innecesarios o mal ubicados significa un costo fijo adicional que podría evitarse; a su vez, la carencia de instrumentos en zonas críticas puede ser peligrosa.

Los datos de planta pueden analizarse empleando modelos matemáticos de estado estacionario que representen adecuadamente el funcionamiento de la planta. Los modelos basados en principios fundamentales están conformados por sistemas de ecuaciones algebraicas **E** que representan balances, relaciones termodinámicas y correlaciones experimentales. Dado que un modelo es una abstracción del problema real, es posible generar diversos modelos con distintos niveles de complejidad para una misma planta según las suposiciones y aproximaciones consideradas. Cuanto más simplificado es el modelo, más sencillo es su posterior análisis. Sin embargo, las simplificaciones conllevan una pérdida de precisión que debe ser tomada en cuenta. En líneas generales, podemos decir que un modelo estará bien construido cuando sea suficientemente detallado como para representar todo el comportamiento de interés y a la vez no posea términos o ecuaciones que compliquen innecesariamente el tratamiento matemático sin contribuir significativamente a los resultados. Los modelos más rigurosos para plantas de procesos son no lineales e involucran balances de masa y energía, así como también relaciones termodinámicas para estimar propiedades tales como densidades, entalpías y constantes de equilibrio.

En términos matemáticos, un *modelo de estado estacionario* de una planta de procesos basada en principios fundamentales está conformado por un sistema de ecuaciones algebraicas no lineales  $E$ , las cuales representan los balances de masa y energía, y las relaciones termodinámicas empleadas para estimar propiedades tales como densidades, entalpías y constantes de equilibrio.

En el área de monitoreo, los *algoritmos de clasificación de variables* constituyen una herramienta fundamental para diseño y revamp de instrumentación. Mediante un análisis apropiado, es posible por ejemplo establecer si la instrumentación existente en una planta es suficiente para conocer todas las variables de interés o bien identificar mediciones redundantes. Básicamente, se pueden diferenciar cuatro categorías de variables según su factibilidad de cálculo (Romagnoli y Sánchez, 1999):

1. **Variables Redundantes:** variables medidas que pueden ser calculadas a partir de los balances y del resto de las variables medidas.
2. **Variables No Redundantes:** variables medidas que no pueden computarse a partir de los balances y de las restantes variables medidas.
3. **Variables Observables:** variables no medidas que pueden obtenerse a partir de las variables medidas usando las ecuaciones de balance.
4. **Variables No Observables:** variables no medidas que no pueden calcularse a partir de las variables medidas mediante las ecuaciones de balance.

Por otra parte, las ecuaciones de  $E$  pueden clasificarse en tres grupos:

1. **Ecuaciones Asignadas:** aquellas que se emplean para despejar variables observables.

2. **Ecuaciones Redundantes:** son aquellas cuyas variables son todas observables o medidas, y que no se emplean para despejar variables observables.
3. **Ecuaciones No Asignadas:** son aquellas que contienen al menos una variable no observable y no pueden emplearse para despejar variables observables.

La tarea de clasificación de variables puede subdividirse en dos fases: *análisis de observabilidad* y *detección de redundancias*. En la primera se clasifican las variables no medidas en observables y no observables, mientras que en la segunda se determinan las mediciones redundantes. En este punto, resulta importante aclarar que el desarrollo de esta tesis se centra en el diseño e implementación de nuevas técnicas que se emplean exclusivamente para realizar el análisis de observabilidad. La determinación de redundancias constituye otra etapa del diseño de instrumentación que requeriría en sí un análisis específico, extenso y profundo que no constituye el objetivo de esta tesis. En este capítulo se revisa el “estado del arte” en el tema al momento de iniciar nuestras investigaciones. Las fallas detectadas en las metodologías existentes constituyeron la principal motivación para desarrollar este trabajo de tesis.

Con respecto a los objetivos propios y específicos del análisis de observabilidad, es posible afirmar que cuando la meta es reducir la cantidad de sensores, resulta factible definir que la mejor clasificación de variables no medidas es aquella que logra determinar el mayor número de variables observables para un conjunto de mediciones preestablecido. En tal sentido, es posible asociar esta definición con un objetivo económico: a menor número de mediciones, más bajos resultarán tanto los costos de inversión como los costos operativos requeridos para lograr el funcionamiento confiable de la planta. Las inversiones se reducen debido a que se evitan mediciones innecesarias; mientras que los costos operativos disminuyen al tener que monitorear una menor cantidad de instrumentos.

Para alcanzar esta meta es esencial utilizar una metodología de clasificación de variables que exhiba las siguientes propiedades:

1. **Eficacia:** capacidad para determinar el mayor número de variables observables a partir de un conjunto preestablecido de mediciones.
2. **Eficiencia:** potencial para obtener la clasificación en tiempos de ejecución razonablemente bajos para problemas industriales de gran dimensión.
3. **Amplio rango de aplicación:** posibilidad de emplear la metodología de clasificación para distintos tipos de plantas, independientemente de las características físicas del proceso y del tipo de modelo matemático usado para representar su funcionamiento.

Teniendo en mente estas premisas, hemos efectuado en lo que resta de este capítulo un estudio crítico de las principales técnicas de clasificación existentes en la literatura a fin de determinar sus ventajas y limitaciones.

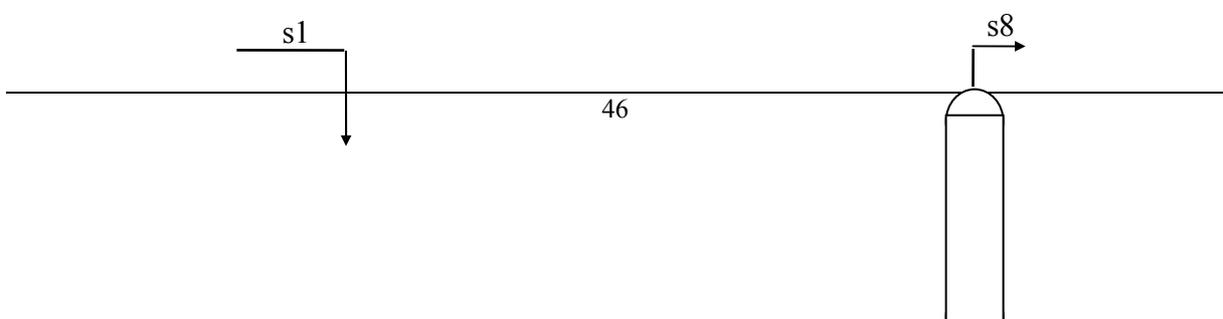
### 3.2. METODOLOGÍAS DE CLASIFICACIÓN

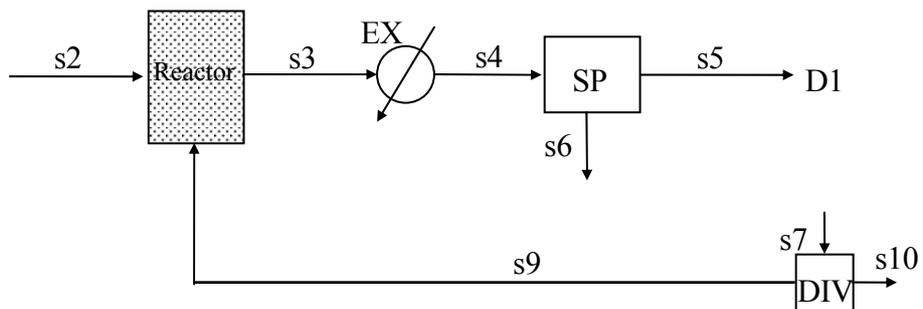
Durante las últimas tres décadas, se han desarrollado dos líneas principales de investigación para efectuar la clasificación de variables en problemas de instrumentación. En la primera, denominada *enfoque orientado a topología*, las variables son clasificadas analizando los ciclos y conjuntos de corte (*cutsets*) que surgen en una serie de grafos derivada del grafo no dirigido subyacente a la topología de la planta. La otra forma de abordar el problema es conocida como *enfoque orientado a ecuaciones*, dado que emplea diferentes matrices asociadas al sistema de ecuaciones  $\mathbf{E}$  que se ha elegido como modelo del proceso.

### 3.2.1. ENFOQUE ORIENTADO A TOPOLOGÍA

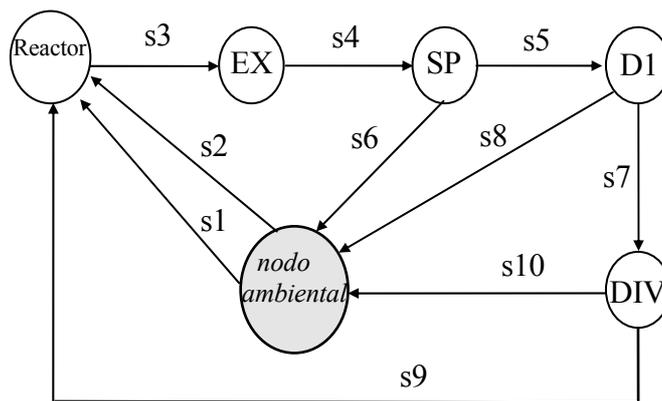
Estos métodos emplean el grafo no dirigido  $G$  subyacente al digrafo generalizado que representa la topología del proceso, es decir la distribución de los equipos y las corrientes que los conectan. La figura 3.1 a) muestra una representación esquemática de la topología de un proceso, conocida como diagrama de flujo (flowsheet). Estos esquemas consisten de símbolos para representar los distintos equipos de procesos y de líneas que representan las cañerías que conectan distintas unidades. Estas líneas se denominan comúnmente corrientes y están caracterizadas por un caudal de materia, composición de la mezcla circulante, temperatura y presión. Los nodos y aristas del digrafo generalizado corresponden a los equipos y corrientes de la planta respectivamente, tal como se muestra en la figura 3.1 b). El digrafo generalizado contiene el diagrama de flujo del proceso y un nodo adicional, denominado nodo del medio ambiente, el cual representa los alrededores de la planta. Todas las alimentaciones a la planta parten de este nodo, mientras que todas las corrientes de productos que salen de la planta ingresan al mismo. A partir de este digrafo se construye el grafo subyacente  $G$ , eliminando los sentidos de las aristas – ver figura 3.1.c).

Vaclavek (1969) fue el primero en usar esta representación. Para la clasificación de redundancias propuso una regla muy simple: unir todos los pares de nodos conectados a través de una corriente (arista) que contenga al menos una variable no medida. De este modo, el grafo  $G'$  resultante sólo involucra mediciones redundantes. Para el análisis de observabilidad, se eliminan de  $G$  todas las corrientes conocidas completamente, es decir, aquellas con todas sus variables medidas, obteniéndose otro grafo reducido  $G''$ . Luego, se buscan todos los ciclos fundamentales en  $G''$ . Los caudales asociados con corrientes ausentes en estos ciclos se clasifican como observables.

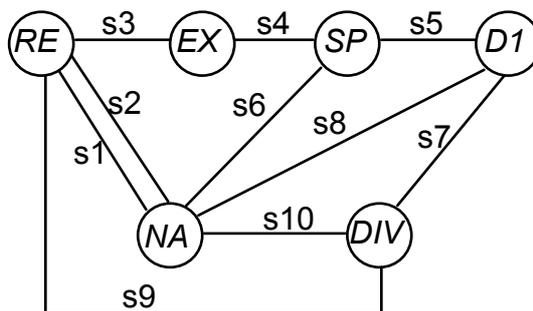




a) Diagrama de flujo de un proceso



b) Digrafo generalizado.



c) Grafo  $G$  subyacente al diagrama de flujo.

**Figura 3.1.** Diagrama de flujo de un proceso y su grafo  $G$  subyacente.

La técnica anterior es adecuada para el tratamiento de relaciones algebraicas lineales, tales como las que surgen de los balances de masa globales. Sin embargo, cuando se incorporan expresiones matemáticas más complejas, tales como balances de masa por componentes o términos vinculados a reacciones químicas, el sistema de ecuaciones  $E$  correspondiente se vuelve fuertemente no lineal. Posteriormente, Vaclavek (1976) extendió su metodología para

considerar relaciones bilineales. No obstante, la nueva técnica exige que, para cada corriente, todas o ninguna de sus fracciones en masa estén medidas, sin permitir situaciones intermedias a pesar de que usualmente en los procesos industriales reales sólo se miden algunas composiciones.

En contraste, Kretsovalis y Mah (1987) introdujeron un algoritmo libre de restricciones con respecto a las mediciones de composición. En dicho trabajo se propuso un procedimiento de clasificación basado en la construcción de varios grafos auxiliares a partir del grafo no dirigido subyacente a la topología del proceso. Luego, mediante la aplicación de una serie de reglas que analizan los ciclos y cutsets de dichos grafos se logra la clasificación de variables y mediciones. Un aspecto novedoso de esta técnica es la inclusión de balances en torno a clusters de nodos, donde cada cluster está conformado por un conjunto relacionado de equipos. También fueron incorporados balances de energía, considerándose una relación unívoca entre la temperatura de una corriente y su entalpía específica.

A pesar de estas mejoras, esta metodología no resulta aplicable en forma directa ante la presencia de reacciones químicas o equipos tipo "splitter". Por esta razón, Kretsovalis y Mah (1988a, 1988b) presentaron posteriormente nuevos algoritmos para clasificación, denominados GENOBS y GENRED, junto con una familia de teoremas que sustentan dichas técnicas. GENOBS clasifica las variables en observables y no observables, mientras que GENRED determina qué mediciones son redundantes. Una estrategia de clasificación similar se propone en Meyer y *co.* (1993). En general, los nuevos métodos incorporan el tratamiento de relaciones bilineales extendiendo el rango de aplicación de la técnica propuesta originalmente por Vaclavek.

Aunque estas nuevas variantes probaron ser muy eficientes, sólo fueron diseñadas para ser aplicadas sobre modelos matemáticos lineales y bilineales. Por tal motivo, estos métodos

pierden rigurosidad en el tratamiento de procesos que incluyen equipos complejos con expresiones fuertemente no lineales, tales como reactores y "flashes".

### 3.2.2. ENFOQUE ORIENTADO A ECUACIONES

Los algoritmos que siguen este enfoque emplean las matrices asociadas al sistema de ecuaciones algebraicas  $\mathbf{E}$  que modela la planta. Se pueden distinguir dos grandes ramas dentro de los métodos orientados a ecuaciones: *técnicas estructurales* y *técnicas no estructurales*. Los procedimientos no estructurales trabajan en forma numérica a partir de la matriz de coeficientes de  $\mathbf{E}$  ó de otras matrices derivadas de ésta. Por ejemplo, en el algoritmo propuesto por Crowe y *co.* (1983) se construye una matriz especial, denominada matriz de proyección, a partir de la matriz de incidencia de  $\mathbf{E}$ . En cambio, Madron (1992) emplea la matriz de coeficientes asociada a la representación lineal del modelo matemático del proceso. Por otra parte, las técnicas estructurales (Romagnoli y Stephanopoulos, 1980; Joris y Kaliventzeff, 1986) consisten en reordenar la matriz de ocurrencia de  $\mathbf{E}$  apropiadamente.

#### 3.2.2.1. Técnicas no estructurales

Crowe y *co.* (1983) presentaron una metodología para la reconciliación de modelos lineales proponiendo el método de matriz de proyección. Tres años después, esta técnica fue extendida para permitir el tratamiento de sistemas bilineales (Crowe, 1986). Finalmente, sobre la base de este último trabajo, Crowe (1989) propuso un nuevo algoritmo para análisis de observabilidad y redundancia, donde se emplean varias matrices de proyección a fin de obtener la clasificación completa de las variables.

En términos generales, el procedimiento particiona la matriz de incidencia en tres categorías según la aparición y localización de las mediciones. Para el análisis de observabilidad, el autor define y demuestra cuatro lemas que brindan condiciones necesarias

y/o suficientes para establecer si una variable es no observable. Luego, el método inicializa todas las variables como observables y aplica los cuatro lemas hasta individualizar todas las variables no observables. Esta técnica permite ubicar las mediciones en lugares arbitrarios (sin restricciones); también considera reacciones químicas, "splitters" y flujos energéticos puros expresados a través de relaciones bilineales. Asimismo, los balances de energía son incorporados asumiendo una correspondencia uno a uno entre temperatura y energía por unidad de masa.

A grandes rasgos, es posible afirmar que la última técnica diseñada por Crowe (1989) es completa en lo referido a modelos lineales y bilineales. Sin embargo, y tal como acontece con las metodologías anteriores, este método no ha sido extendido para sistemas con ecuaciones fuertemente no lineales y por lo tanto, su aplicación a estos casos se ve muy limitada.

Por su parte, Madron (1992) emplea la matriz de coeficientes aumentada correspondiente al sistema de ecuaciones  $\mathbf{E}$  linealizado. En un paso preliminar, la matriz es permutada de forma tal que los coeficientes de las variables no medidas aparezcan en las primeras columnas. Luego, se procesa la matriz aplicando el método de eliminación de Gauss-Jordan en dos etapas. En la primera sólo se toma en cuenta para el pivoteo la submatriz correspondiente a las variables no medidas, a fin de efectuar el análisis de observabilidad. La segunda etapa realiza la clasificación de redundancias empleando el mismo proceso de eliminación de Gauss-Jordan pero ahora sobre la submatriz de variables medidas.

En principio, este procedimiento únicamente es aplicable a sistemas lineales, lo cual constituye obviamente una fuerte desventaja, dado que la mayoría de los procesos de interés son de naturaleza no lineal. Aunque existen varias maneras de linealizar ecuaciones, tales como expansiones de Taylor, generación de correlaciones lineales y cambios de variables, todas estas técnicas conducen a una pérdida de precisión que deteriora la calidad de los resultados.

Además, el análisis resulta válido en un entorno del punto de operación elegido. Esta es una de las principales razones por las cuales el desarrollo de metodologías no lineales es un tópico de investigación importante en la actualidad.

### 3.2.2.2. *Técnicas estructurales*

Del análisis efectuado en las secciones previas se puede concluir que tanto las técnicas no estructurales orientadas a ecuaciones como los métodos basados en topología han sido diseñados específicamente para modelar relaciones lineales y bilineales. Aunque en general los procedimientos descritos arriba son muy eficientes, todos poseen una inherente pérdida de rigurosidad en el tratamiento de equipos que incluyen expresiones matemáticas altamente no lineales.

En tal sentido, los algoritmos estructurales proveen una mejor alternativa debido a que permiten una mayor independencia del grado de no linealidad de modelo matemático empleado para representar el comportamiento de la planta. Más aún, estas técnicas constituyen un análisis de observabilidad global ya que pueden aplicarse sin necesidad de definir un punto de operación específico. Esto constituye una ventaja pues a veces se posee información poco precisa al respecto o bien existen diversas condiciones operativas de interés. En este enfoque, aspectos no estructurales tales como singularidades y funciones implícitas pueden ser tomados en cuenta mediante la introducción de verificaciones numéricas complementarias.

El desarrollo de esta línea de trabajo comienza con la técnica propuesta por Romagnoli y Stephanopoulos (1980), la cual está estrechamente vinculada con el algoritmo de determinación de orden de precedencia desarrollado por Stadtherr y *co.* (1974). Este método construye una matriz de ocurrencia  $\mathbf{O} = [ \mathbf{N} \mid \mathbf{M} ]$  cuyas primeras columnas corresponden a las variables no medidas del modelo (submatriz  $\mathbf{N}$ ), ubicando las mediciones en las últimas columnas (submatriz  $\mathbf{M}$ ). Luego, se efectúa un particionamiento de la matriz que permite categorizar las

variables no medidas. El reordenamiento estructural obtenido por esta técnica se muestra en la figura 3.2, donde los bloques sombreados indican los sectores que pueden contener elementos no nulos.

El procedimiento clasifica las variables no medidas (correspondientes a las primeras  $nm$  columnas) en observables y no observables, es decir, reordena la submatriz  $N$  de  $(a+r+na) \times nm$ . Una ecuación es asignada a cada variable observable, por lo tanto, en el patrón de reordenamiento,  $a = o$ . El algoritmo obtiene una submatriz de orden  $a$  con una forma triangular inferior en bloques específica, cuyos bloques sobre la diagonal son cuadrados y pueden ser resueltos secuencialmente. Cada uno de estos bloques se denomina *subconjunto de asignación* (*assignment subset*). Tal como se observa en la figura, el subconjunto de ecuaciones redundantes  $r$  comprende algunas ecuaciones que dependen de variables observables y medidas (filas  $rom$ ), mientras que las restantes son función de las mediciones únicamente (filas  $rm$ ). Por último, las filas correspondientes a  $na$  están asociadas a las ecuaciones que contienen variables no observables.

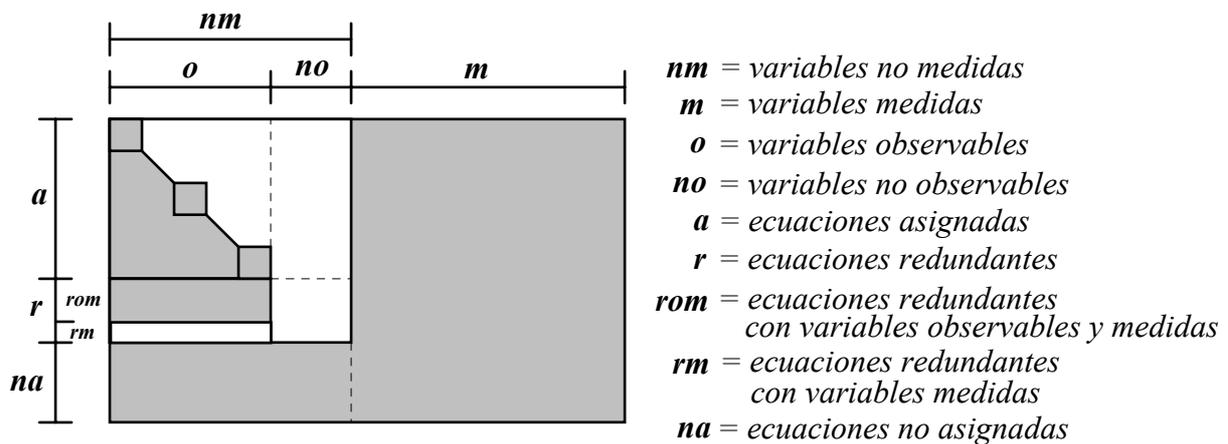


Figura 3.2. Matriz de ocurrencia reordenada obtenida por Romagnoli y Stephanopoulos.

Años después, Joris y Kaliventzeff (1987) diseñaron un reordenamiento estructural completamente diferente, el cual no descompone  $N$  en bloques. El principal problema de este

método es que puede fallar ante la presencia de reciclos debido a que estos bucles usualmente conducen a singularidades numéricas que a veces no son detectadas mediante este análisis estructural básico.

Posteriormente, una versión mejorada de la técnica de clasificación de variables no medidas de Romagnoli y Stephanopoulos fue incluida en PLADAT, un paquete para reconciliación de datos de planta desarrollado por Sánchez y *co.* (1992). En particular, la implementación de ese algoritmo de observabilidad fue denominada **Método de Bloque de Columna Dominante** ó **DCB** (*Dominant Column Block*). Un aspecto importante a tener en cuenta es que el método propuesto originalmente por Romagnoli y Stephanopoulos es aplicable sólo bajo la suposición de que la composición de una corriente es completamente conocida (es decir se miden todas las fracciones molares) o completamente desconocida (no se mide ninguna componente de la corriente). En cambio, DCB no impone tal requerimiento, tornándose así más flexible que su técnica predecesora. Otro aspecto interesante de remarcar es que el particionamiento a forma triangular en bloques facilita la detección de singularidades numéricas, ya que las mismas sólo pueden aparecer en los bloques cuadrados de la diagonal de la submatriz  $axo$ . Estos bloques tienen usualmente una dimensión muy pequeña, razón por la cual el chequeo de la presencia de singularidades numéricas puede realizarse con bajos costos computacionales. En tal sentido, DCB resulta superior al método presentado por Joris y Kaliventzeff.

Aunque DCB es muy eficiente en cuanto a tiempo de ejecución, hemos detectado que esta metodología es poco robusta (Ponzoni y *co.*, 1995). En este contexto el término **robustez** hace referencia a la capacidad del algoritmo para detectar el máximo número de subconjuntos de asignación de mínimo tamaño. En particular, esta técnica falla en la localización de buena parte de los bloques de orden tres en adelante, lo cual implica una potencial pérdida de efectividad

debido a que esta omisión puede conducir a una clasificación errónea en donde algunas variables observables sean categorizadas como no observables.

Del estudio crítico efectuado en este capítulo, se puede concluir que las técnicas estructurales resultan ser las más adecuadas para el análisis de observabilidad, dado su amplio rango de aplicación. No obstante, ninguna de las metodologías hasta aquí mencionadas satisface conjuntamente todos los requerimientos de eficacia, eficiencia, robustez y rango de aplicación, indispensables para el tratamiento riguroso de los modelos matemáticos actualmente utilizados para representar plantas industriales reales. Por tal motivo, en los siguientes capítulos de esta tesis se proponen distintos algoritmos estructurales de observabilidad basados en propiedades bien establecidas de teoría de grafos, que apuntan a superar las dificultades de los métodos descriptos anteriormente.

## ***GS-FLCN: OBSERVABILIDAD Y GRAFOS NO DIRIGIDOS***

### **4.1. INTRODUCCIÓN**

En vista de las falencias encontradas en las técnicas existentes se decidió investigar la posibilidad de diseñar nuevos algoritmos estructurales de observabilidad que fuesen robustos y eficientes en el tratamiento de modelos no lineales. El primero de los métodos propuestos tuvo como origen la estrategia combinatorial seguida por DCB. La nueva metodología (Ponzoni y *co.* 1997; Ponzoni y *co.*, 1999), denominada *Estrategia Global con Primer Nodo Menos Conectado*, en inglés *Global Strategy with First Least-Connected Node (GS-FLCN)*, se basa en la exploración de grafos no dirigidos mediante técnicas de búsqueda en profundidad. Los

grafos explorados se construyen a partir de distintas submatrices correspondientes a diferentes subsistemas del sistema de ecuaciones  $\mathbf{E}$  que modela el proceso.

## 4.2. ASPECTOS BÁSICOS DEL GS-FLCN

En términos generales, GS-FLCN sigue los lineamientos básicos utilizados en DCB. Dada la matriz de ocurrencia  $\mathbf{N}$ , correspondiente a la submatriz  $(a+r+na)*o$  de la figura 3.2, tanto GS-FLCN como DCB aplican la siguiente metodología:

1. La detección de un subconjunto de asignación se efectúa buscando  $n$  columnas de la matriz de ocurrencia  $\mathbf{N}$  tales que su eliminación provoque  $n$  filas vacías. Estas columnas y filas constituyen un subconjunto de asignación de tamaño  $n$ .
2. El procedimiento principal que guía la detección de los subconjuntos de asignación avanza en forma incremental (por tamaño de bloques). Esto implica que primero se buscan todos los subconjuntos de orden 1, luego los de orden 2 y así siguiendo.

La diferencia central entre ambas técnicas radica en la forma en que se llevan adelante estas dos tareas.

### 4.2.1. ESTRATEGIA GLOBAL DE BÚSQUEDA DE SUBCONJUNTOS DE ASIGNACIÓN

El algoritmo 4.1 esquematiza la política de exploración seguida por el procedimiento principal de GS-FLCN, el cual recibe como dato de entrada la matriz de ocurrencia correspondiente al sistema de ecuaciones  $\mathbf{E}$  y retorna como salida la matriz  $\mathbf{R}$  sobre la cual se almacena la solución. En  $\mathbf{R}$ , inicialmente vacía, se ubican los distintos subconjuntos de asignación admisibles hallados por los procedimientos de búsqueda ubicados según el orden en que son detectados.

**Algoritmo 4.1.** Estrategia global para detección de subconjuntos de asignación (**GS-FLCN**).

Datos de entrada: Matriz de ocurrencia de **E**

Datos de salida: **R**

1. Construcción de la matriz de ocurrencia **N**.
  2. Triangularización hacia adelante (**N,R**).
  3.  $n = 2$ 
    - 3.1 Construcción de la Submatriz de Ocurrencia( $n, \mathbf{S}$ ).
    - 3.2 Subrutina 2 (**S,R**).  
(localización de subconjuntos de asignación de  $2 \times 2$ )  
**Si** se detecta un subconjunto admisible,  
    **entonces** volver al paso 2.  
    **si no** ir al paso 4.  
    **fin-si**
  4.  $n = 3$ 
    - 4.1 Construcción de la Submatriz de Ocurrencia ( $n, \mathbf{S}$ ).
    - 4.2 Algoritmo Modificado ( $n, \mathbf{S}, \mathbf{R}$ )  
(localización de subconjuntos de asignación de  $3 \times 3$ )  
**Si** se detecta un subconjunto admisible,  
    **entonces** volver al paso 2.  
    **si no** ir al paso 5.  
    **fin-si**
  5.  $n = 4$ 
    - 5.1 Construcción de la Submatriz de Ocurrencia ( $n, \mathbf{S}$ ).
    - 5.2 Aplicar el paso 1 de la Subrutina 2.
    - 5.3 Algoritmo FLCN ( $n, \mathbf{S}, \mathbf{R}$ )  
(localización de subconjuntos de asignación de  $n \times n$ ,  $n \geq 4$ )  
**Si** se detecta un subconjunto admisible,  
    **entonces** volver al paso 2.  
    **si no**  
        **Si** ( $n >$  máximo tamaño de subconjunto),  
            **entonces** Terminar y retornar la clasificación;  
            **si no**  $n = n+1$  y volver al paso 5.1.  
        **fin-si**
- fin-si**

---

El primer paso del algoritmo consiste en construir la matriz de ocurrencia **N**, cuyas filas y columnas representan las ecuaciones y variables no medidas de **E** respectivamente. **N** posee un elemento no nulo  $n_{ij}$  si y solo si la  $j$ -ésima variable aparece en la  $i$ -ésima ecuación. En la siguiente etapa, conocida como *Triangularización hacia adelante*, se realiza la detección de todos los subconjuntos de asignación de orden 1. Esta subrutina busca filas que contengan un único elemento no nulo. Cada una de estas filas representa una ecuación  $e$  que posee una única incógnita  $x$ , razón por la cual  $x$  puede ser despejada directamente de  $e$ . Una vez detectado un

---

bloque, la fila y columna que lo conforman son removidas de  $\mathbf{N}$ . La eliminación de las columnas que integran los subconjuntos de orden 1 puede provocar que otras filas reduzcan su cantidad de elementos no nulos. Por lo tanto, la triangulación hacia adelante se repite hasta que para un recorrido completo de  $\mathbf{N}$  no se detecte ningún nuevo subconjunto de orden 1.

La subrutina ***Construcción de la submatriz de ocurrencia*** construye submatrices  $\mathbf{S}$  de  $\mathbf{N}$  con el fin de reducir el espacio de búsqueda y así hacer más eficiente la detección de subconjuntos de asignación. Esta tarea consta de dos pasos:

1. Seleccionar todas las filas de  $\mathbf{N}$  que contengan a lo sumo  $n$  elementos no nulos junto con las columnas en las que aparezcan dichos elementos y ponerlas en  $\mathbf{S}$ .
2. Borrar de  $\mathbf{S}$  las columnas con un único elemento no nulo. Luego, remover aquellas filas que hayan quedado con un sólo elemento no nulo. Repetir estos pasos hasta que todas las filas y columnas de  $\mathbf{S}$  tengan al menos dos elementos no nulos.

La utilización de  $\mathbf{S}$  en lugar de  $\mathbf{N}$  hace que la búsqueda se ejecute únicamente sobre aquellas columnas que potencialmente pueden conformar un subconjunto de asignación de orden  $n$ . Es importante destacar que la reducción de  $\mathbf{N}$  a  $\mathbf{S}$  preserva todos los subconjuntos de tamaño  $n$ , lo cual se establece en el siguiente teorema.

**Teorema 3.1.** Todo bloque de orden  $n$ , cuadrado y estructuralmente no singular, contenido en la matriz de ocurrencia  $\mathbf{N}$  también está presente en la submatriz reducida  $\mathbf{S}$ .

*Demostración.* Supongamos por el contrario que existe un bloque  $\mathbf{B}$  de orden  $n$  de  $\mathbf{N}$  que no está contenido en  $\mathbf{S}$ . Luego, al menos una de las filas ó columnas de  $\mathbf{B}$  no está en  $\mathbf{S}$ . Consideremos en primer término que una columna  $c$  de  $\mathbf{B}$  no está en  $\mathbf{S}$ . Si este es el caso, entonces  $c$  tiene que haber sido asignada anteriormente. Pero, si  $c$  fue asignada, tampoco puede estar en  $\mathbf{N}$ . Luego, el hecho que  $c$  esté en  $\mathbf{N}$  pero no en  $\mathbf{S}$  es un absurdo. Analicemos ahora el

otro caso: que falte una fila  $f$  de  $\mathbf{B}$  en  $\mathbf{S}$ . En tal caso, por definición de  $\mathbf{S}$ , la fila  $f$  tiene que contener más de  $n$  elementos. Pero esto implica que  $f$  no puede pertenecer a  $\mathbf{B}$ , dado que la remoción de las  $n$  columnas de  $\mathbf{B}$  no dejaría vacía a  $f$ . Esto es un absurdo que provino de suponer que  $f$  no estaba en  $\mathbf{S}$ . Por lo tanto toda fila de  $\mathbf{B}$  está en  $\mathbf{S}$ . Finalmente, se puede concluir que todas las filas y columnas de un bloque de  $\mathbf{N}$  también están presentes en  $\mathbf{S}$ .

Los procedimientos *Subrutina 2*, *Algoritmo Modificado* y *FLCN* efectúan la detección de los bloques de orden 2, 3 y  $n$  ( $n \geq 4$ ) respectivamente. Como paso preliminar a la aplicación de cualquiera de estos algoritmos, se debe generar la submatriz  $\mathbf{S}$  correspondiente al tamaño del subconjunto a buscar. Cada vez que se detecta un nuevo subconjunto, este debe pasar un *test de admisión* a fin de descartar todos aquellos bloques que correspondan a subsistemas numéricamente singulares.

En tal sentido, algunas restricciones para admitir un subconjunto son generadas automáticamente por el software, en tanto que otras condiciones de prohibición son establecidas manualmente por el ingeniero de procesos. Algunos subsistemas con ecuaciones redundantes, tales como los formados por corrientes paralelas o por todos los balances de masa por componente de una corriente más su balance de masa global, pueden ser prohibidos directamente por el software. Otros tipos de restricciones, que no responden a un patrón fácil de automatizar, deben ser especificadas por el ingeniero.

Si el bloque supera este test, sus filas y columnas son removidas de  $\mathbf{N}$  y se transfiere el control al procedimiento de triangularización hacia adelante. El retorno a esta subrutina se debe a que la remoción de un subconjunto puede generar el desacople de nuevos subsistemas de orden 1. Por otro lado, si el bloque no es admisible, se continúa la búsqueda de subsistemas de tamaño  $n$  hasta detectar un bloque válido o agotar todas las combinaciones factibles. En este último caso, se inicia la detección de subconjuntos de dimensión  $n+1$ . Tal como se muestra en

el algoritmo 4.1, la búsqueda avanza por tamaño de subconjunto, siendo necesario construir la submatriz  $S$  para cada nueva dimensión a explorar.

La Subrutina 2 es similar a la propuesta por Stadtherr *y co.* (1974). A grandes rasgos, este procedimiento busca dos columnas que posean dos filas en común a fin de conformar un bloque de orden 2. El algoritmo 4.2 presenta la manera en que se seleccionan estas filas y columnas.

---

**Algoritmo 4.2.** Subrutina 2.

*Datos de entrada:*  $S$

*Datos de salida:*  $R$

1. **Si**  $S$  tiene al menos dos columnas, **entonces**  
     Buscar la columna  $c_1$  de  $S$  con mayor número de elementos no nulos. Si hay más de una columna en esta situación, elegir la de menor denominación.  
     **si no** Terminar.  
     **fin-si.**
  2. Eliminar  $c_1$  de  $S$ , junto con todas las filas que tienen elementos no nulos en  $c_1$ .
  3. Buscar una columna  $c_2$  cuyo número de elementos no nulos se haya reducido en dos luego de aplicar el paso 2.
  4. Formar un bloque  $B$  de orden dos con  $c_1$ ,  $c_2$  y las dos filas presentes en ambas columnas.
  5. **Si**  $B$  aprueba el test de admisión, **entonces**  
     Eliminar  $c_2$  de  $S$ . Ubicar las filas y columnas de  $B$  en las primeras dos filas y columnas libres de la matriz reordenada  $R$ . Eliminar de la matriz  $M$  todas las filas y columnas en  $B$ . Terminar.  
     **si no** Volver al paso 1.  
     **fin-si.**
- 

El Algoritmo Modificado fue diseñado inicialmente como una mejora de la *Subrutina N* (Stadtherr *y co.*, 1974) para la detección de bloques de orden 3 en adelante. Ambos procedimientos construyen los bloques en forma combinatorial. El algoritmo 4.3 describe los pasos seguidos por el Algoritmo Modificado. La principal diferencia con el método de Stadtherr está en el paso 3, en donde el Algoritmo Modificado sólo exige la existencia de  $n-1$  columnas cuyo número de elementos no nulos haya sido reducido al menos en uno, mientras que la Subrutina N eleva este último requisito a dos o más elementos. De este modo, la Subrutina N

explora menos cantidad de combinaciones que el Algoritmo Modificado, lo cual mejora la eficiencia en detrimento de la robustez del método ya que puede no detectarse un número importante de subconjuntos de asignación admisibles. En contrapartida, el Algoritmo Modificado propuesto logra una mejor exploración del espacio de búsqueda sin incrementos significativos en los tiempos de cómputo.

Aunque el Algoritmo Modificado logra un mejor desempeño que el procedimiento propuesto por Stadtherr, el método puede fallar en la detección de bloques de tamaño 4 en adelante. Todas las columnas de los subconjuntos estructuralmente no singulares de orden 3 poseen al menos dos elementos no nulos en común con las restantes columnas que forman el bloque. Por tal motivo el Algoritmo Modificado es infalible en la detección de los subconjuntos de este tamaño. Sin embargo, los bloques válidos de orden 4 o mayor pueden no contar con una columna base que posea dos filas en común con las otras columnas. Luego, el método puede no encontrar algunos subconjuntos de asignación correctos. Esto nos motivó a desarrollar otra técnica para estos casos, la cual fue denominada *First Least-Connected Node (FLCN)*, que sigue una política más efectiva en la construcción de las combinaciones que conforman los subconjuntos de asignación.

#### Algoritmo 4.3. Algoritmo Modificado

*Datos de entrada:* **S**

*Datos de salida:* **R**

1. **Si** **S** tiene al menos  $n$  columnas,  
**entonces**  
     Buscar la columna  $c_1$  de **S** con mayor número de elementos no nulos. Si hay más de una columna en esta situación, elegir la de menor denominación.  
     **si no** Terminar.  
     **fin-si.**
2. Eliminar  $c_1$  de **S**, junto con todas las filas que tienen elementos no nulos en  $c_1$ .
3. **Si** existen  $n-1$  columnas cuyo número de elementos no nulos ha sido reducido en uno o más,  
**entonces**

---

```

Remove temporalmente de S las n-1 columnas seleccionadas.
Si n filas quedaron sin elementos,
    entonces
        Remove temporalmente de S dichas n filas.
        Construir un bloque B de orden n con las filas
        y columnas borradas.
        Ir al paso 4.
    si no
        Restaurar las columnas que habían sido removidas
        temporalmente.
        Volver al paso 1.
    fin-si.
si no Volver al paso 1.
fin-si.
4. Si el bloque B es admisible,
    entonces
        Ubicar las filas y columnas de B en las primeras n filas
        y columnas libres de la matriz reordenada R.
        Eliminar de la matriz N todas las filas y columnas en B.
        Terminar.
    si no
        Recobrar las filas y columnas que habían sido borradas
        temporalmente de S.
        Volver al paso 1.
fin si.

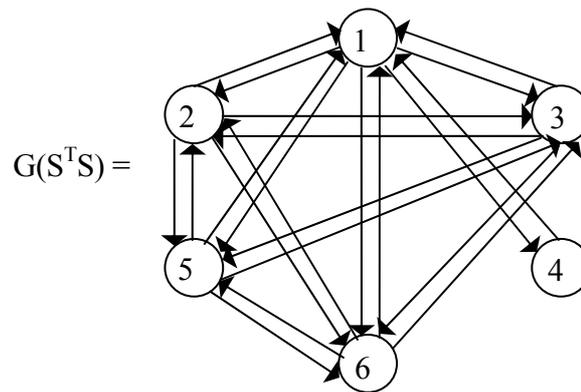
```

---

#### 4.2.2. PRIMER NODO MENOS CONECTADO (FIRST LEAST-CONNECTED NODE)

La subrutina FLCN explora el grafo no dirigido  $G$  correspondiente a la matriz de incidencia de  $S^T S$ , siendo  $S$  la submatriz de  $N$  obtenida en el paso 5.1 de la estrategia global. Los nodos de  $G$  representan a todas las variables del sistema de ecuaciones  $E$  contenidas en el subsistema de ecuaciones correspondiente a la submatriz  $S$ . Por su parte, las aristas relacionan a las variables que aparecen en una misma ecuación. En términos de  $S$ , existirá una arista entre los nodos  $n_i$  y  $n_j$  de  $G$  si y sólo si existe al menos una fila en  $S$  tal que contiene elementos no nulos en las  $i$ -ésima y  $j$ -ésima columnas. En la figura 4.1 se muestra un ejemplo.

$$\mathbf{S} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 \end{bmatrix} \quad \mathbf{S}^T \mathbf{S} = \begin{bmatrix} 2 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 0 & 1 & 2 \\ 1 & 1 & 2 & 0 & 2 & 1 \\ 1 & 0 & 0 & 2 & 0 & 0 \\ 1 & 1 & 2 & 0 & 2 & 1 \\ 1 & 2 & 1 & 0 & 1 & 2 \end{bmatrix}$$



**Figura 4.1.** Submatriz  $\mathbf{S}$  y digrafo  $\mathbf{G}$  asociado a  $\mathbf{S}^T \mathbf{S}$ .

FLCN construye las combinaciones de filas y columnas que pueden conformar un subconjunto de asignación factible mediante la exploración del grafo  $\mathbf{G}$ . Para detectar bloques de tamaño  $n$  el algoritmo efectúa una búsqueda en profundidad (DFS) de todos los caminos de longitud  $n-1$  presentes en  $\mathbf{G}$ . Cada uno de estos caminos representa una combinación factible de  $n$  variables. Más precisamente, cada nodo del camino tiene asociado una variable; luego, las columnas de  $\mathbf{S}$  asociadas a los  $n$  nodos de un camino constituyen una combinación de  $n$  variables relacionadas entre sí por distintas ecuaciones.

La elección de los nodos a visitar durante el recorrido sigue una heurística que da nombre a la técnica. Esta heurística prioriza la exploración de los nodos menos conectados del grafo. Es importante destacar que, al igual que los algoritmos orientados a topología, GS-FLCN involucra el recorrido de grafos. Sin embargo, las técnicas orientadas a topología son conceptualmente distintas de GS-FLCN. Las primeras usan el grafo subyacente a la topología del proceso, mientras que GS-FLCN emplea otro grafo, el cual representa la relación estructural entre las

filas y columnas de la submatriz de ocurrencia **S**. En otras palabras, GS-FLCN es orientado a ecuaciones dado que el grafo que explora refleja la relación existente entre las variables y ecuaciones de **E**. Una descripción recursiva del método se presenta en el algoritmo 4.4.

En este punto, resulta importante aclarar que el análisis de observabilidad usualmente requiere varias ejecuciones de la estrategia global descrita en el algoritmo 4.1. En general, una vez obtenida una primera clasificación de las variables no medidas; los resultados son analizados por el ingeniero de procesos a fin de establecer si todas las variables de interés son observables o no. A consecuencia de esto, el usuario puede decidir agregar o remover instrumentos en la planta o incorporar nuevas restricciones. En tales casos, resulta necesario volver a efectuar el análisis de observabilidad para la nueva configuración de la planta. Este procesamiento se repetirá hasta alcanzar una clasificación de las variables no medidas que sea completamente satisfactoria desde el punto de vista del usuario.

---

**Algoritmo 4.4.** First Least-Connected Node (**FLCN**).

*Datos de entrada:* **S**, **n**, **nodo**, **camino**, **longitud**  
*Datos de salida:* **éxito**, **R**

```

Si longitud = 0,
  entonces
    Si n filas de S quedaron vacías,
      entonces
        Construir un conjunto de asignación B con las n filas
        vacías y las n columnas asociadas a los nodos del camino.
        Aplicar el test de admisión a B.
        Si B es admisible,
          entonces
            Remover de N las filas y columnas de B.
            Agregar las filas y columnas de B a las
            primeras n filas y columnas libres
            de la matriz reordenada R. éxito = verdadero.
          si no éxito = falso.

```

```

    fin-si
    si no éxito = falso.
  fin-si
si no
  finalizar = falso.
  Mientras (no finalizar) y (no éxito) hacer:
    [*]Elegir un nodo sin estampilla adyacente a nodo.
    Si existen varios nodos adyacentes sin marcar,
    seleccionar el menos conectado. Si hay varios nodos con
    el menor nivel de conexión elegir el de menor etiqueta.
    Si no existe un nodo adyacente a nodo sin marcar,
    entonces
      finalizar = verdadero. éxito = falso.
    si no
      nodo-anterior = nodo.
      nodo = el nodo elegido en [*]. Marcar el nodo.
      FLCN (S,n,nodo,camino,longitud-1,éxito,R)
      Si no éxito,
        entonces
          Remove nodo del camino.
          nodo = nodo-anterior.
    fin-si
  fin-si
  fin-mientras
fin-si

```

---

### 4.3. COMPARACIÓN TEÓRICA ENTRE DCB Y GS-FLCN

La diferencia básica entre DCB y GS-FLCN radica en la cantidad de combinaciones exploradas y en la forma en que se construyen las combinaciones para subconjuntos de orden tres en adelante. Como se explicó anteriormente, DCB emplea la Subrutina N como una extensión natural de la Subrutina 2, mientras que GS-FLCN utiliza dos procedimientos diferentes, el Algoritmo Modificado y FLCN. En principio, el Algoritmo Modificado podría ser aplicado para la detección de subconjuntos de cualquier orden, sin embargo, sólo posee desempeño óptimo para los bloques de tamaño 3, perdiendo robustez en los restantes casos. Por tal motivo, se propone el uso del algoritmo First Least-Connected Node (FLCN), el cual sigue un enfoque diferente, para subconjuntos de orden mayor a tres.

FLCN y DCB pueden ser comparados en términos de **S**. Por un lado, cuando DCB busca subconjuntos de tamaño  $n$ , selecciona una columna  $c_1$  entre aquellas con mayor número de elementos no nulos. Luego, se construyen combinaciones de  $n-1$  columnas  $c_2, c_3, \dots, c_n$ , cada una de las cuales estará asociadas con al menos dos filas  $i_{p1}$  y  $i_{p2}$  tales que:

$$\mathbf{S}(i_{p1}, c_1) = \mathbf{S}(i_{p1}, c_p) = 1 \quad \text{y} \quad \mathbf{S}(i_{p2}, c_1) = \mathbf{S}(i_{p2}, c_p) = 1 \quad \text{para } p = 2, \dots, n. \quad (4.1)$$

Si se encuentra un conjunto de columnas que satisfacen estos requerimientos, la columna  $c_1$  es eliminada y el procedimiento se repite. De este modo, DCB omite explorar muchos potenciales subconjuntos de asignación. Al imponer la presencia de una columna base  $c_1$  fuertemente asociada al resto de las columnas (debido a que la combinación se construye en torno a ella) se restringe la posibilidad de detectar bloques raros cuyos nodos (variables) estén menos conectados.

En cambio, FLCN construye combinaciones de  $n$  columnas  $c_1, c_2, \dots, c_n$  tales que  $c_p$ , para  $1 < p < n$ , está en al menos dos filas  $i_{p1}, i_{p2}$  las cuales verifican:

$$\mathbf{S}(i_{p1}, c_{p-1}) = \mathbf{S}(i_{p1}, c_p) = 1 \quad \text{y} \quad \mathbf{S}(i_{p2}, c_p) = \mathbf{S}(i_{p2}, c_{p+1}) = 1 \quad \text{para } p = 2, \dots, n-1. \quad (4.2)$$

Con la siguiente condición para los extremos:

$$\mathbf{S}(i_{11}, c_1) = \mathbf{S}(i_{11}, c_n) = 1$$

Esta metodología es más robusta que la anterior ya que, al no requerirse una columna base, se puede construir un mayor número de combinaciones. Esto obviamente permite detectar más subconjuntos de asignación. A modo de ilustración considérese el bloque de la figura 4.2 extraído de un ejemplo presentado por Pissanetzky (1984).

Mientras que el subconjunto de  $7 \times 7$  es detectado por GS-FLCN, DCB falla al intentar localizarlo debido a que ninguna de las siete columnas puede ser tomada como columna base. Por ejemplo, si la columna 1 es elegida como base, las columnas 3, 5, 6 y 7 no poseen filas en

común con ella. Por lo tanto, estas no son tomadas como posibles miembros de un subconjunto de asignación. El mismo problema acontece con la elección de cualquier otra columna base.

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

*Figura 4.2. Subconjunto del ejemplo de Pissanetzky detectado por GS-FLCN.*

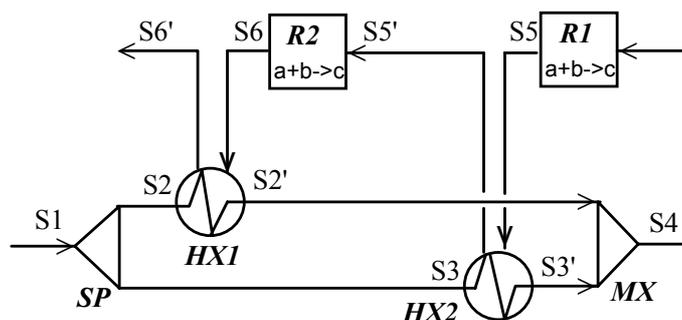
Además, la política de trabajo de DCB no es apropiada si el principal objetivo es obtener el mayor número posible de bloques de mínimo tamaño. Después de construir la submatriz, los procedimientos Subrutina 2 y N de DCB buscan todos los posibles bloques de tamaño  $2 \times 2$  o  $n \times n$  respectivamente, sin volver hacia atrás para chequear la aparición de nuevos subconjuntos de menor orden. Este retroceso es aconsejable por razones de robustez dado que frecuentemente la remoción de un bloque de causa el desacople de subconjuntos de asignación más pequeños, los cuales no son detectados usando DCB. En contraste, GS-FLCN siempre conduce a bloques irreducibles que aseguran subconjuntos de mínimo tamaño.

#### **4.4. DCB VERSUS GS-FLCN: APLICACIONES INDUSTRIALES**

Ambas estrategias fueron empleadas para resolver varios problemas industriales a fin de establecer comparaciones de desempeño. En esta sección se presentan dos ejemplos que revelan claramente las ventajas de GS-FLCN frente a DCB.

#### 4.4.1. CASO DE ESTUDIO I: EJEMPLO ACADÉMICO

Este problema, presentado por Joris y Kalitventzeff (1986), corresponde a una sección de planta conformada por dos reactores R1 y R2, dos intercambiadores de calor HX1 y HX2, un separador SP y un mezclador MX. La figura 4.3 muestra un esquema del proceso.



**Figura 4.3.** Diagrama de flujo del proceso (Joris y Kalitventzeff, 1986).

La tabla 4.1 lista las variables del proceso, el cual contiene 3 componentes (a, b, c) y 10 corrientes (1, 2, 2', 3, 3', 4, 5, 5', 6, 6'). La letra T se emplea para indicar temperaturas, Fr para caudales y U para coeficientes de transferencia de calor.

Variables							
1	a S1	7	T S5'	13	U R1	19	T S3
2	b S1	8	T S6	14	a S6	20	Fr S2
3	c S1	9	T S6'	15	b S6	21	Fr S3
4	T S1	10	a S5	16	c S6	22	T S2'
5	T S4	11	b S5	17	U R2	23	T S3'
6	T S5	12	c S5	18	T S2		

**Tabla 4.1.** Variables del modelo (Joris & Kalitventzeff, 1986).

En este caso, DCB únicamente asigna dos subconjuntos de orden correspondientes a las ecuaciones 9 y 10 y las variables T S2 y T S3 respectivamente. En cambio, GS-FLCN logra encontrar todos los bloques reportados por Joris y Kalitventzeff. Los resultados obtenidos por GS-FLCN se muestran en la tabla 4.2, donde los subconjuntos detectados aparecen sombreados. Es interesante notar que los subconjuntos de tamaño 4 son muy raros. DCB no pudo encontrarlos debido a que las columnas de dichos bloques están débilmente conectadas.

	VARIABLES MEDIDAS									VARIABLES NO MEDIDAS													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
R1 balance de masa comp. a	1									1			1										
R1 balance de masa comp. b		1									1		1										
R1 balance de masa comp. c			1									1	1										
R1 balance de energía	1	1	1		1	1				1	1	1											
R2 balance de masa comp. a										1				1			1						
R2 balance de masa comp. b											1				1		1						
R2 balance de masa comp. c												1				1	1						
R2 balance de energía							1	1		1	1	1		1	1	1							
SP balance de energía 1				1															1				
SP balance de energía 2				1																1			
SP balance de masa																					1	1	
HTX1 balance de energía	1	1	1					1	1					1	1	1		1		1		1	
HTX2 balance de energía	1	1	1			1	1			1	1	1							1		1		1
MX balance de energía	1	1	1		1															1	1	1	1

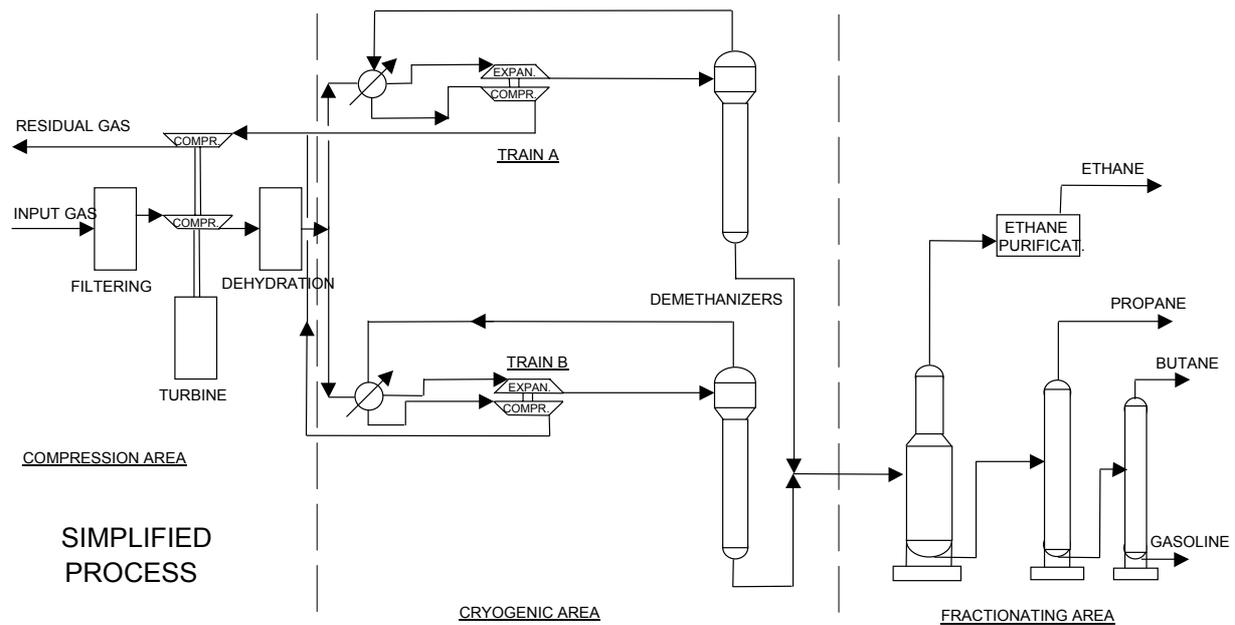
**Tabla 4.2.** Conjuntos de asignación encontrados por GS-FLCN para el caso de estudio I.

#### 4.4.2. CASO DE ESTUDIO II: PLANTA INDUSTRIAL DE PRODUCCIÓN DE ETANO.

En este ejemplo se presenta el análisis de observabilidad de una planta que separa etano de gas natural.

##### 4.4.2.1. Descripción de la Planta

En la figura 4.4 se muestra un esquema general del diagrama de flujo. El proceso puede ser dividido en tres secciones principales: compresión de gas y deshidratación, separación criogénica y fraccionamiento. El estudio involucra 87 unidades enlazadas por 185 corrientes que contienen 12 componentes. La tabla 4.3 contiene una lista de las unidades.



**Figura 4.4.** Esquema simplificado de la planta de producción de etano.

Tipo de Equipo	Cantidad
<i>Secadores</i>	16
<i>Bombas isotérmicas</i>	8
<i>Columnas de destilación</i>	6
<i>Condensadores parciales</i>	1
<i>Compresores</i>	10
<i>Divisores</i>	9
<i>Unidades generales</i>	2
<i>Intercambiadores de calor</i>	10
<i>Mezcladores</i>	7
<i>Rebullidores</i>	4
<i>Separadores</i>	2
<i>Separadores isotérmicos</i>	6
<i>Turbinas</i>	3
<i>Válvulas</i>	3

**Tabla 4.3.** Lista de las unidades del proceso.

El tamaño de la matriz de ocurrencia está estrechamente relacionado con el nivel de detalle considerado para el modelo. En este ejemplo, el conjunto de ecuaciones del modelo incluye balances de masa y energía, así como también relaciones termodinámicas para densidades, entalpías y constantes de equilibrio. También se incluyeron mediciones de

caudales, composiciones, temperaturas y presiones. El modelo resultante contiene 1830 ecuaciones y 1425 variables no medidas. Dada la magnitud y complejidad del problema, tanto el sistema de ecuaciones como la matriz de ocurrencia correspondientes al modelo y algunas restricciones necesarias para el test de admisión fueron generados automáticamente utilizando el software para construcción de modelos denominado ModGen (Vazquez y *co.*, 2001).

La tabla 4.4 presenta el número de conjuntos de asignación obtenidos por cada algoritmo. La primera columna indica el tamaño de los bloques detectados. La última fila se refiere al *porcentaje de variables asignadas (pva)*, calculado con respecto al total de variables no medidas.

Tamaño de bloques	DCB	GS-FLCN
1	715	897
2	4	7
3	5	5
4	1	1
5	-	1
6	-	3
7	-	2
19	-	2
<i>pva</i>	52,07 %	70,53 %

**Tabla 4.4.** Resultados para la planta de producción de etano.

#### 4.4.2.2. Factores de ramificación

Los resultados reportados para GS-FLCN en el caso de estudio II fueron obtenidos utilizando *factores de ramificación (branching factors, BF)* como técnica de aceleración. El espacio de búsqueda explorado por GS-FLCN es una expansión forestal donde los nodos y aristas pertenecen a  $G(S^T S)$ . La técnica de aceleración consiste en podar algunas ramas de los árboles con el objeto de reducir el tiempo de búsqueda. De este modo, es posible alcanzar los niveles más profundos de búsqueda de forma más rápida, mejorando el desempeño del

algoritmo para casos de estudio reales de gran dimensión. En particular, nuestra implementación permite al usuario indicar los BF para cada nivel, lo cual permite concentrar la búsqueda en aquellos sectores que a priori parezcan más prometedores.

#### 4.4.3. ANÁLISIS DE LOS RESULTADOS

GS-FLCN exhibe definitivamente un mejor desempeño en cuanto a robustez que DCB, dado que encuentra una mayor cantidad de subconjuntos de asignación. La aparición de más subconjuntos de asignación también conduce a un incremento en la cantidad de variables asignadas, constituyendo una notable mejora en cuanto a efectividad. Esto resulta claro si se comparan los métodos en términos de los índices de *pva*. Además, la tabla 4.4 revela que GS-FLCN exhibe un mejor desempeño en cuanto a la calidad de la descomposición del sistema dado que detecta un número mayor de bloques de menor dimensión.

Otro aspecto a destacar es que, aunque ambas utilizan las mismas estrategias para buscar los subconjuntos de orden 1 y 2, GS-FLCN encuentra un 25% más de bloques de estos tamaños. La razón para este incremento radica en que cada vez que un nuevo bloque es hallado, GS-FLCN recomienza la búsqueda para subconjuntos de tamaño 1. Así, se observa experimentalmente que el aislamiento de un bloque frecuentemente conduce a posteriores desacoples de subconjuntos más pequeños.

Cabe remarcar que los subconjuntos de asignación encontrados por ambas estrategias coinciden al principio; sólo difiere el orden en que se detectan. Luego, en cierto punto, GS-FLCN halla un bloque de 5x5 que DCB no puede detectar. Dicho subconjunto se muestra en la figura 4.5, donde se puede apreciar que no contiene una columna base  $c_1$  que satisfaga los requerimientos impuestos en (4.1). Esta es la principal diferencia entre los resultados.

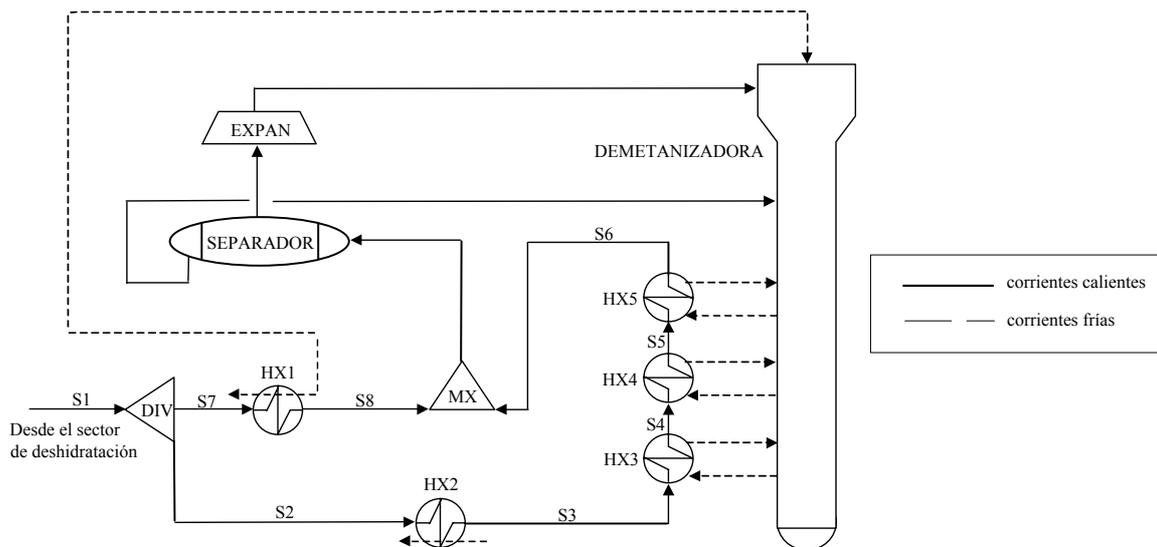
$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

**Figura 4.5.** Subconjunto de asignación de tamaño 5.

Luego, GS-FLCN encuentra dos bloques de orden 7, los cuales nunca hubieran sido detectados por DCB porque no contienen ninguna columna base. Ambos subconjuntos poseen igual patrón de realidad, ya que corresponden a un grupo de balances de masa en torno de los mismos equipos pero para diferentes componentes (en un caso se trata de propano y en el otro de dióxido de carbono). El sector de planta mostrado en la figura 4.6 es básicamente un esquema de integración energética. El ciclo detectado por el algoritmo, el cual comprende las corrientes S1 a S9, involucra a las corrientes calientes de una serie de intercambiadores de calor que evaporan líquido retirado desde una demetanzadora a diferentes puntos, con un reciclo que retorna al nivel inferior de la columna.

#### 4.4.4. TEST DE ADMISIÓN EMPLEADO EN GS-FLCN

GS-FLCN siempre detecta bloques estructuralmente no singulares, pero los subsistemas de ecuaciones asociados a los conjuntos de asignación necesitan tener solución numérica unívoca para que las variables clasificadas como observables puedan ser calculadas a través de las ecuaciones a las que fueron asignadas. La estrategia GS-FLCN presentada en este capítulo es inherentemente estructural. De esto se desprende la necesidad de diseñar un test de admisión sobre la base de conceptos de observabilidad global, es decir, un chequeo que no exija asignar valores específicos a las variables del proceso dado que en muchas ocasiones no se cuenta con dicha información o los valores disponibles son imprecisos y por ende poco confiables.



**Figura 4.6.** Grupo de intercambiadores de calor en el área criogénica.

equipo / corriente	$x_{S2}$	$x_{S3}$	$x_{S4}$	$x_{S5}$	$x_{S6}$	$x_{S8}$	$x_{S7}$
HX2	1	1	0	0	0	0	0
HX3	0	1	1	0	0	0	0
HX4	0	0	1	1	0	0	0
HX5	0	0	0	1	1	0	0
MX	0	0	0	0	1	1	0
HX1	0	0	0	0	0	1	1
DIV	1	0	0	0	0	0	1

**Figura 4.7.** Subconjunto de asignación del ciclo S1-S9 de la figura 3.6.

El procedimiento de clasificación es aplicado iterativamente. Después de cada iteración, el ingeniero analiza los resultados a fin de chequear que todos los subsistemas aislados tengan solución única. Para esto se efectúa un análisis conceptual y/o simbólico que permita determinar los bloques que deben ser prohibidos. En muchos casos, una simple inspección visual es suficiente. Una vez que un subconjunto ha sido detectado, el ingeniero introduce esta información al programa como una restricción adicional y vuelve a comenzar una nueva iteración. A partir de este punto, cada vez que un bloque **B** es encontrado, el programa lo confronta con su lista de restricciones. Si algún subconjunto prohibido está incluido o es igual a

**B**, el bloque **B** es descartado como subconjunto válido y la búsqueda sigue explorando. En caso contrario **B** es removido de **S** e incorporado a la solución.

Más aún, la observación cuidadosa de la estructura de la matriz puede ayudar al diseñador a encontrar la ubicación más conveniente para una dada medición. Supongamos, por ejemplo, que se hubiese encontrado un bloque numéricamente singular con el patrón de ralidad de la figura 4.7 (esto es hipotético dado que para el caso de estudio II el bloque de  $7 \times 7$  tiene solución única). Si la primera columna fuera removida mediante la medición de  $x_{S2}$ , las restantes variables asociadas al bloque podrían ser totalmente desacopladas en subconjuntos de asignación de orden 1 mediante la aplicación de la triangularización hacia delante. Por lo tanto, el agregado de una única medición sería suficiente para que todas las variables del bloque se volvieran observables.

Otro aspecto a tener en cuenta es la presencia de subsistemas cuya resolución numérica o simbólica es compleja. Los bloques correspondientes a ecuaciones de balances de masa lineales, tales como el mostrado en la figura 4.7, pueden ser resueltos en forma sencilla. Sin embargo, cuando el subsistema contiene ecuaciones que involucran funcionalidades implícitas, su tratamiento posterior puede resultar muy difícil. Por esta razón muchas veces resulta conveniente agregar unas pocas mediciones en los lugares precisos a fin de simplificar los cálculos. En tales casos, observar la estructura del bloque obtenido con GS-FLCN ayuda a decidir la mejor localización para los instrumentos adicionales.

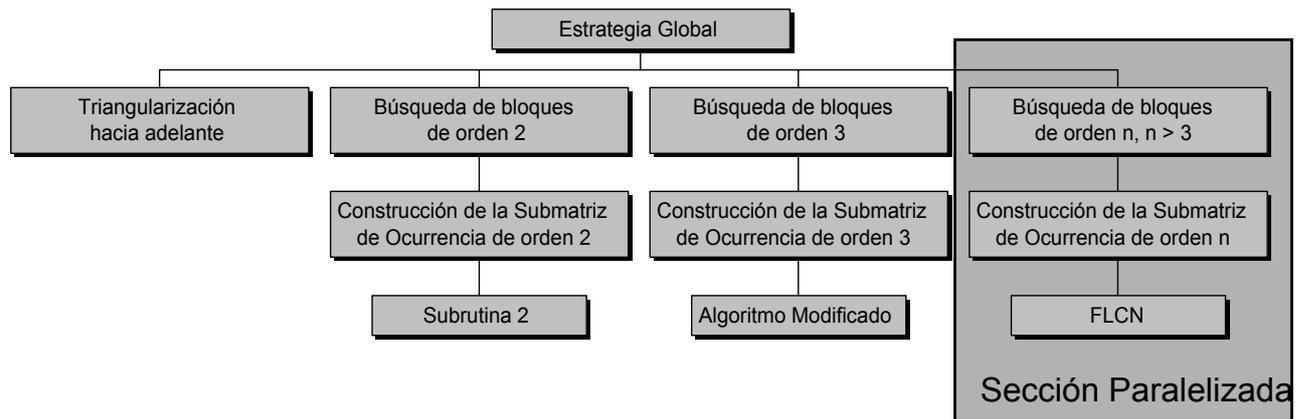
## 4.5. GS-PFLCN: UN ALGORITMO PARALELO-DISTRIBUIDO

El procesamiento paralelo es una herramienta que permite una reducción significativa en lo referido a tiempos de ejecución en aplicaciones con alto consumo de tiempo. El enfoque paralelo tradicional promueve el desarrollo de algoritmos para computadoras paralelas (Anderson y *co.*, 1995). Sin embargo, la gran inversión económica requerida para adquirir este tipo de arquitecturas de cómputo y la falta de estándares existente entre los fabricantes dificultan estos desarrollos. Por tal motivo, la tendencia actual es "migrar" el concepto de procesamiento paralelo hacia ambientes distribuidos heterogéneos (Freund y Siegel, 1993; Khokhar y *co.*, 1993; Crowl L., 1994). El desarrollo de algoritmos paralelos diseñados para entornos de procesamiento distribuido resulta mucho más viable dado que sólo es necesario contar con una *red de área local* (*local area network, LAN*) de estaciones de trabajo. Esta configuración ofrece bajos costos de inversión inicial y fácil escalabilidad del sistema.

La naturaleza combinatorial de GS-FLCN hizo que la paralelización de la técnica resultará factible y muy atractiva en cuanto a ganancias en tiempo de ejecución. Por otra parte, dada la infraestructura disponible en nuestro ámbito de trabajo y las razones expuestas anteriormente, se optó por el procesamiento paralelo-distribuido. La implementación paralela del algoritmo fue denominada GS-pFLCN.

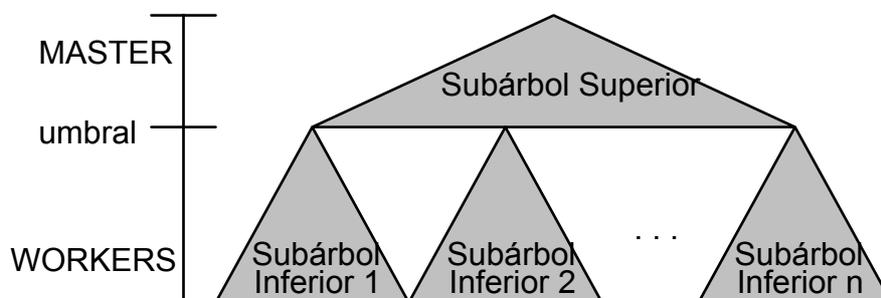
### 4.5.1. PARALELIZACIÓN DE GS-FLCN

La estructura de GS-FLCN sigue el esquema mostrado en la figura 4.8. La búsqueda de subconjuntos de asignación de orden 4 en adelante es la sección que consume la mayor parte del tiempo de cómputo. Por tal motivo, se consideró conveniente paralelizar esta parte del código, mientras que los restantes procedimientos fueron preservados en sus versiones secuenciales.



**Figura 4.8.** Estructura General de GS-FLCN y sección paralelizada.

La subrutina principal de esta sección es FLCN. Tal como se mencionó anteriormente, FLCN efectúa una DFS generando una expansión forestal. Una estrategia simple para paralelizar estas búsquedas consiste en partir el espacio de búsqueda cuando la DFS supera cierto umbral o nivel (*threshold*) en el árbol de expansión que está explorando. De este modo, todos los subárboles que nacen a partir de dicho umbral pueden ser procesados en paralelo, tal como se muestra en la figura 4.9.



**Figura 4.9.** Descomposición del espacio de búsqueda.

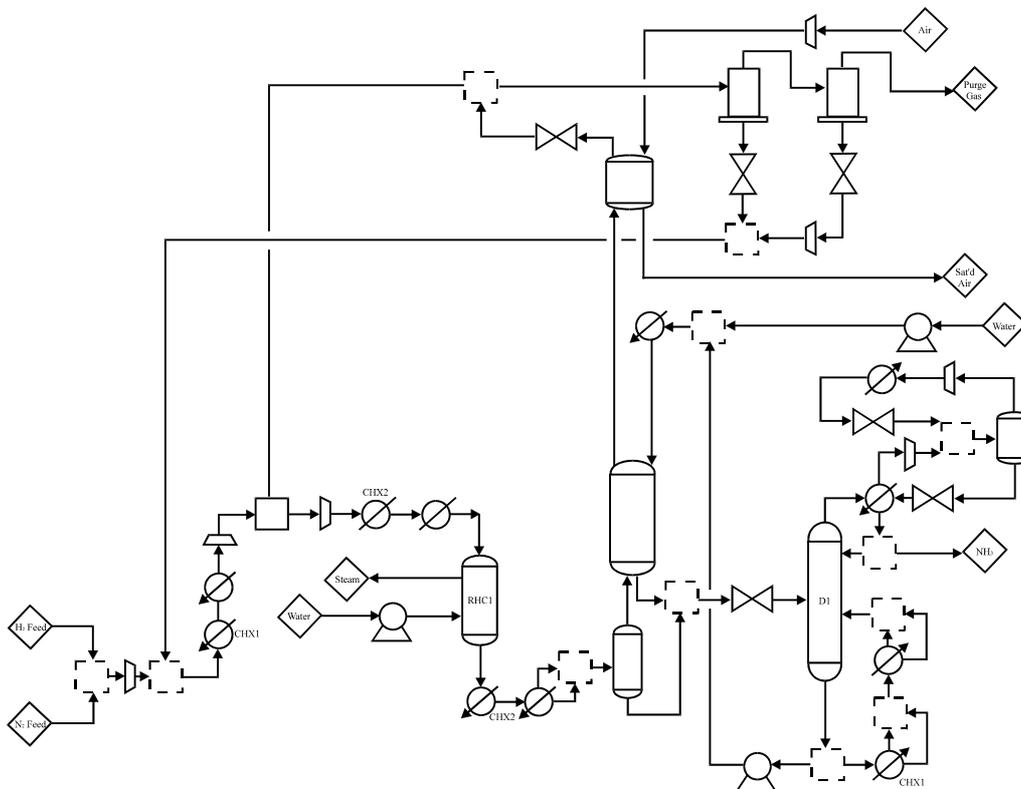
Este enfoque sigue un modelo MASTER-WORKER donde el MASTER explora el subárbol superior y distribuye los subárboles inferiores entre los WORKERS, quienes efectúan la exploración de los mismos. GS-pFLCN (Ponzoni *et al.*, 2001) fue implementado en lenguaje C usando la librería de pasaje de mensajes Parallel Virtual Machine (PVM).

#### 4.5.2. ANÁLISIS DEL DESPEÑO DE GS-PFLCN

GS-pFLCN fue aplicado a la clasificación de variables no medidas para ecuaciones de balance de masa y energía de dos plantas de procesos reales.

##### 4.5.2.1. Casos de estudio industriales

El primer ejemplo elegido fue una planta de síntesis de amoníaco diseñada por Bike y *co.* (1985) para la producción 1500 toneladas diarias de amoníaco mediante el proceso de síntesis denominado Haber-Bosch. El diagrama de flujo de la planta se presenta en la figura 4.10. El modelo matemático no lineal que representa está planta consta de 560 ecuaciones algebraicas y 516 variables no medidas. El segundo caso de estudio propuesto fue la planta de producción de etano analizada anteriormente. El diagrama de flujo fue mostrado en la figura 4.4; y el modelo matemático usado es fuertemente no lineal y posee 1830 ecuaciones y 1425 variables medidas.



**Figura 4.10.** Diagrama de flujo de la planta de síntesis de amoníaco.

#### 4.5.2.2. Resultados del análisis de observabilidad efectuado con GS-pFLCN

Las corridas del programa fueron realizadas sobre un ambiente distribuido homogéneo conformado por 10 computadoras Pentium de 200 MHz con sistema operativo LINUX interconectadas mediante una red de área local Ethernet. Para alcanzar óptimos resultados la red fue aislada de otros procesos que pudieran interferir con las corridas. El desempeño del GS-pFLCN fue contrastado con su versión secuencial observándose importantes ganancias en tiempos de ejecución. La tabla 4.5 muestra los tiempos consumidos por las corridas secuenciales y paralelas. Se aprecia que la reducción de los tiempos resultó muy significativa para ambos casos de estudio.

La ganancia de tiempos lograda por GS-pFLCN se vuelve más significativa aún si se toma en cuenta que el algoritmo de observabilidad es ejecutado iterativamente varias veces hasta alcanzar una clasificación de variables completamente satisfactoria.

Número de Procesadores	Tiempo transcurrido (elapsed time) en minutos	
	Planta de Amoníaco	Planta de Etano
1 (secuencial)	34:19	58:23
2	18:47	36:54
4	09:51	21:54
6	06:58	17:05
8	05:42	14:25
10	04:53	11:53

**Tabla 4.5.** Tiempos de ejecución de GS-pFLCN.

Finalmente, la tabla 4.6 cuantifica la ganancia obtenida en términos de speed-up y eficiencia, dos métricas comúnmente empleadas en procesamiento paralelo. Ambas establecen relaciones entre el tiempo secuencial y el paralelo teniendo en cuenta la cantidad de procesadores empleados.

La cota superior para el valor de speed-up, conocido como speed-up lineal, es igual a la cantidad de procesadores empleados y corresponde a una eficiencia del 100 %. Este límite representa una situación ideal donde los costos de comunicación son despreciables y las unidades de procesamiento nunca están ociosas. Dada las características de los ambientes de procesamiento reales, estos valores nunca alcanzan en la práctica dicha cota superior.

Número de Procesadores	Speed-Up / Eficiencia	
	Planta de Amoníaco	Planta de Etano
2	1.82 / 91%	1.58 / 79 %
4	3.42 / 87%	2.64 / 66 %
6	4.92 / 82%	3.41 / 56 %
8	6.02 / 75%	4.05 / 50 %
10	7.02 / 70%	4.91 / 49 %

**Tabla 4.6.** Speed-Up y Eficiencia logrados con GS-pFLCN.

De los resultados obtenidos puede concluirse que las ganancias en tiempo fueron muy satisfactorias, más aún si se tiene en cuenta que la red de área local es de 10 Mb. Mejores resultados podrían alcanzarse sobre LANs más rápidas.

#### 4.5.2.3. *Ganancias en términos de robustez*

La introducción del paralelismo en la estrategia GS-FLCN hace posible explorar un mayor espacio de búsqueda con tiempos de cómputo razonables. En otras palabras, se pueden elegir valores más altos para los factores de ramificación. En general, se recomienda aumentar los valores de BF porque de esta forma se reduce el riesgo de omitir subconjuntos de asignación factibles durante la búsqueda. Por ejemplo, en la tabla 4.7 se presentan los resultados obtenidos para la planta de síntesis de amoníaco utilizando diferentes valores de BF. La tabla muestra la cantidad de subconjuntos de asignación de cada tamaño encontrados con BF=3 y BF=5 para el nivel de profundidad 6. Aunque la cantidad de variables asignadas es la misma, la calidad del particionamiento es mejor para la segunda corrida. Este se debe a que al aumentar la densidad

de exploración en el nivel 6, otro bloque de orden 6 es detectado. A su vez, la remoción de este bloque ocasiona el desacople de dos bloques de tamaño 1. De hecho, el bloque de tamaño 8 de la primera corrida es reducible y desaparece cuando la búsqueda es más exhaustiva.

<i>Tamaño del bloque</i>	<i>Número de Bloques</i>	
	<i>Corrida 1 (BF=3)</i>	<i>Corrida 2 (BF=5)</i>
1	297	299
2	6	6
3	2	2
4	1	1
6	3	4
7	1	1
8	1	-
11	2	2

**Tabla 4.7.** Corridas de GS-pFLCN para la planta de síntesis de amoníaco con diferentes BF.

## 4.6. CONCLUSIONES

En este capítulo se presentó un nuevo algoritmo estructural para realizar el análisis de observabilidad en plantas de procesos. La técnica propuesta, denominada Global Strategy with First Least-Connected Node (GS-FLCN) clasifica las variables no medidas mediante un reordenamiento estructural de la matriz de ocurrencia correspondiente al modelo matemático empleado para representar el funcionamiento de una planta. La permutación se realiza mediante la búsqueda incremental de subconjuntos de asignación, los cuales están asociados a las variables observables. Una de las principales características de esta nueva técnica es su capacidad para manejar sistemas altamente no lineales. Además, la incorporación de chequeos de admisión permite tener en cuenta restricciones generales que van más allá de un mero análisis estructural.

Se realizó un estudio comparativo entre GS-FLCN y DCB a fin de evaluar el desempeño del método propuesto. Este análisis involucró comparaciones en aspectos teóricos, así como también la confrontación de resultados obtenidos para diferentes casos de estudio industriales. GS-FLCN resultó ser mucho más robusto y eficiente que su predecesor, logrando determinar un mayor número de conjuntos de asignación y variables observables.

Dada su naturaleza combinatorial, la nueva metodología resulta computacionalmente costosa en cuanto a tiempos de ejecución para problemas que contienen subconjuntos de asignación de gran dimensión, los cuales requieren niveles de búsqueda más profundos. Ante esta situación, el uso de factores de ramificación resultó muy satisfactorio al permitir centrar la exploración en aquellas regiones del espacio de búsqueda con mayores probabilidades de contener subconjuntos de asignación.

Otra alternativa para acelerar los tiempos de cómputo fue la implementación de una versión paralelo-distribuida del algoritmo que denominamos GS-pFLCN. El método fue paralelizado siguiendo un modelo MASTER-WORKER y el principio de descomposición de dominios sobre el espacio de búsqueda. Los resultados alcanzados con esta implementación fueron excelentes, permitiendo ahorros muy significativos en cuanto a tiempo de ejecución. Más aún, la metodología paralelizada permite obtener resultados más confiables dado que permite una mayor exploración del espacio de soluciones, lo cual incrementa el grado de robustez.

Otro aspecto de interés de GS-FLCN es que la descomposición estructural obtenida por el método conduce a una configuración con una mínima cantidad de instrumentos, permitiendo a su vez una fácil visualización de los lugares más convenientes para adicionar o remover sensores. Esto conduce a su vez a importantes ahorros en inversiones y costos operativos.

Por todo lo expresado anteriormente, se concluye que GS-FLCN resulta una estrategia confiable, robusta y eficiente para la clasificación de variables en problemas de instrumentación de plantas de procesos. Asimismo, GS-FLCN posee un amplio rango de aplicación gracias a su enfoque estructural, pudiendo ser aplicada en modelos matemáticos fuertemente no lineales.

## *CDHG: OBSERVABILIDAD BASADA EN HIPERGRAFOS*

### **5.1. INTRODUCCIÓN**

En el capítulo anterior se describió detalladamente el método GS-FLCN para análisis de observabilidad, donde se demostró que esta nueva técnica es mucho más robusta que sus predecesoras. No obstante, el método puro, es decir sin factores de ramificación (BF), posee un tiempo de ejecución que crece considerablemente a medida que aumenta el tamaño de los subconjuntos a explorar. Dado que esto constituía una seria desventaja, se consideró necesario desarrollar nuevas alternativas. Una posibilidad consistía en diseñar variantes de GS-FLCN que

aceleraran el proceso de búsqueda. Este es sin duda el objetivo que motivó el desarrollo de los factores de ramificación y el GS-pFLCN presentados en el capítulo anterior.

La otra alternativa fue el diseño de algoritmos estructurales para el análisis de observabilidad utilizando un enfoque no combinatorio, aún cuando esto implicase una cierta pérdida de robustez para algunos casos. De este modo, se propuso un nuevo algoritmo no combinatorio (Ponzoni y co., 1998a), denominado *Detección de Ciclos en Hipergrafos* (*Cycle Detection on Hypergraphs, CDHG*), el cual describiremos a lo largo del presente capítulo.

## 5.2. CDHG: UN ALGORITMO BASADO EN HIPERGRAFOS

El algoritmo propuesto está basado en los conceptos de hipergrafos enunciados en el capítulo 2. La idea central es que, asociando un hipergrafo  $\mathbf{H}$  a la matriz de ocurrencia  $\mathbf{N}$  correspondiente a las ecuaciones y variables no medidas del sistema de ecuaciones  $\mathbf{E}$  que modela la planta, se pueden individualizar los subconjuntos de asignación de  $\mathbf{N}$  mediante la detección de ciclos en  $\mathbf{H}$ .

### 5.2.1. ALGORITMO BÁSICO

El algoritmo 5.1 muestra la secuencia de pasos realizada por CDHG. En primer lugar, se asocia un hipergrafo  $\mathbf{H}(\mathbf{N}) = (\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_r)$  a la matriz de ocurrencia  $\mathbf{N}$ , cuya dimensión es  $r \times c$  de modo tal que las columnas (variables) y las filas (ecuaciones) estén asociadas con los nodos  $n_i$  y las aristas  $\mathbf{A}_j$  en  $\mathbf{H}(\mathbf{N})$ , respectivamente.

---

**Algoritmo 5.1.** Detección de Ciclos en Hipergrafos (CDHG).

```

Datos de entrada:  $\mathbf{H}(\mathbf{N})$ 
Datos de salida:  $\mathbf{H}(\mathbf{R})$ 

Haux  $\leftarrow \mathbf{H}(\mathbf{N})$ ;  $\mathbf{H}(\mathbf{R}) \leftarrow \emptyset$ ;
ord  $\leftarrow 0$ ;
finaliza  $\leftarrow$  falso;
Mientras no finaliza hacer:
  Si no EncuentraAutoAristas(Haux, ord,  $\mathbf{H}(\mathbf{R})$ )
    entonces
      heurística  $\leftarrow$  primera heurística;
      Mientras no EncuentraHGP(Haux, heurística, ord,  $\mathbf{H}(\mathbf{R})$ ) y
        no finaliza hacer:
        Si Ultima(heurística)
          entonces finaliza  $\leftarrow$  verdadero.
          si no heurística  $\leftarrow$  Próxima(heurística).
        fin-si
      fin-mientras
    fin-si
  fin-mientras

```

Una vez que se ha establecido esta relación, se inicializa **Haux** con  $\mathbf{H}(\mathbf{N})$ ; **Haux** es un hipergrafo auxiliar que será reducido cada vez que se encuentre un subconjunto de asignación.  $\mathbf{H}(\mathbf{R})$  se inicializa en  $\emptyset$ ;  $\mathbf{H}(\mathbf{R})$  es el hipergrafo que corresponde a la matriz de ocurrencia reordenada  $\mathbf{R}$ . Este hipergrafo se va construyendo a medida que se detectan los subconjuntos. Finalmente, ord corresponde al orden de  $\mathbf{H}(\mathbf{R})$ . Al comienzo, ord es igual a 0. Luego, CDHG trata de encontrar subconjuntos de tamaño uno por medio de la función *EncuentraAutoAristas* presentada en el algoritmo 5.2.

Esta subrutina examina todas las aristas de **Haux** en forma secuencial. Siempre que se encuentra una autoarista  $\mathbf{A}_i = \{n_j\}$ , se la remueve de **Haux** y se la incorpora a  $\mathbf{H}(\mathbf{R})$ , de modo tal que  $\mathbf{A}_i$  y  $n_j$  se conviertan en la (ord+1)-ésima arista y el (ord+1)-ésimo nodo de  $\mathbf{H}(\mathbf{R})$  respectivamente. Este procedimiento se repite varias veces hasta que no se detecte ninguna nueva autoarista en una barrida completa. La política es iterativa porque cada vez que se remueve una autoarista, una arista ya explorada puede convertirse en una autoarista si contenía el vértice de la autoarista eliminada. Es interesante destacar que esta función es equivalente a la subrutina de triangularización hacia adelante usado en otros algoritmos de particionamiento.

**Algoritmo 5.2.** Función **EncuentraAutoAristas**.

---

*Datos de entrada-salida:* **Haux**, ord, **H(R)**  
*Valor retornado por la función:*  
verdadero: si encuentra al menos una autoarista,  
falso: en caso contrario.

éxito  $\leftarrow$  falso.  
aristasVisitadas  $\leftarrow \emptyset$ .  
**Mientras** queden aristas de **Haux** sin visitar **hacer**:  
  arista  $\leftarrow$  una arista de **Haux** no visitada.  
  aristasVisitadas  $\leftarrow$  aristasVisitadas  $\cup$  {arista}.  
  **Si** la cardinalidad de arista es 1  
    **entonces** {arista es una autoarista}  
    nodo  $\leftarrow$  Nodo(arista, **Haux**).  
    **Si** Chequeo1x1(nodo, arista, **Haux**)  
      **entonces**  
      ord  $\leftarrow$  ord+1;  
      AgregarHG1x1(nodo, arista, ord, **H(R)**).  
      RemoveArista(arista, **Haux**).  
      RemoveNodo(nodo, **Haux**).  
      éxito  $\leftarrow$  verdadero.  
    **fin-si**  
  **fin-si**  
**fin-mientras**  
**Retornar** el valor de éxito.

---

Cuando no pueden hallarse más autoaristas, CDHG intenta detectar subconjuntos de tamaño mayor o igual que 2 mediante la subrutina **BuscarHGP** descrita en el algoritmo 5.3. Esta subrutina realiza una búsqueda en profundidad (DFS) a lo largo del hipergrafo **Haux** siguiendo reglas heurísticas. Las reglas diseñadas serán explicadas más adelante, en este mismo capítulo. **BuscarHGP** hace uso de dos pilas para efectuar la búsqueda: una para los nodos y otra para las aristas. Las pilas son estructuras de datos comúnmente usadas en algoritmos de búsqueda. Una descripción clara y detallada de los conceptos principales relacionados con pilas puede encontrarse en Aho y co. (1983). En nuestro método, las pilas actúan como memorias que almacenan el camino ya explorado. De este modo, el tope de la pila contiene el último nodo visitado, denominado nodo actual  $n_a$ . Siempre que se efectúa un movimiento hacia un nodo  $n_p$  adyacente a  $n_a$  a través de una arista  $A_p$ ,  $n_p$  y  $A_p$  son empujadas hacia arriba en las pilas para

nodos y aristas respectivamente. Entonces,  $n_p$  constituye el nuevo nodo actual  $n_a$ . Una vez que todos los nodos adyacentes al nodo actual  $n_a$  han sido explorados,  $n_a$  se remueve de la pila y el procedimiento continúa con el nodo que ahora aparece al tope de la pila. Este procedimiento continúa hasta que se vacíe la pila ó se detecte un ciclo con un subconjunto permitido.

---

### Algoritmo 5.3. Función **BuscarHGP**.

*Datos de entrada:* heurística, Restricciones

*Datos de entrada-salida:* **Haux**, ord, **H(R)**

*Valor retornado por la función:*

verdadero: si encuentra un hipergrafo parcial,  
falso: en caso contrario.

éxito  $\leftarrow$  falso.

ok  $\leftarrow$  verdadero.

pila\_nodos  $\leftarrow$  pila\_vacia.

pila\_aristas  $\leftarrow$  pila\_vacia.

InicializarEnFalso(estampilla).

InicializarEnFalso(visitado).

InicializarEnFalso(aristas\_exp).

**Mientras** ok **hacer:**

ok  $\leftarrow$  BuscarNodo(**Haux**,heurística,visitado,nodo);

**Si** ok **entonces** {se ha encontrado un nodo no visitado}

Poner el nodo en la pila\_nodos. éxito  $\leftarrow$  falso.

estampilla[nodo]  $\leftarrow$  verdadero. visitado[nodo]  $\leftarrow$  verdadero.

**Mientras** (no hubo éxito)**y**(no este vacía pila\_nodos) **hacer:**

existe  $\leftarrow$  BuscarNodoAdyacente(nodo,**Haux**,heurística,  
estampilla,aristas\_exp,nodo\_ady,arista).

**Si** existe **entonces** {Se encontró un nodo adyacente}

Poner la arista en la pila\_aristas.

aristas\_exp[arista]  $\leftarrow$  verdadero.

**Si** estampilla[nodo\_ady]

**entonces** {nodo\_ady estaba en la pila,  
luego existe un ciclo}

**Si** TestCiclo(**Haux**,nodo\_ady,pila\_nodos,  
pila\_aristas,Restricciones,**PH**,n)

**entonces** {Se halló un hipergrafo parcial}

ok  $\leftarrow$  falso. éxito  $\leftarrow$  verdadero.

AgregarEnHGNxN(**PH**,n,ord,**H(R)**);

**si no** {este subconjunto es inadmisibile}

aristas\_exp[Tope(pila\_aristas)]  $\leftarrow$  falso.

Desapila pila\_aristas.

**fin-si**

**si no** {nodo\_ady no está sobre la pila}

Poner nodo\_ady en la pila\_nodos.

nodo  $\leftarrow$  nodo\_ady. estampilla[nodo]  $\leftarrow$  verdadero.

visitado[nodo]  $\leftarrow$  verdadero.

**fin-si**

**si no** {No se encontró un nodo adyacente}

```

    aristas_exp[Tope(pila_aristas)] ← falso.
    Desapila pila_aristas.
    Si no está vacía pila_nodos entonces
        aristas_exp[Tope(pila_aristas)] ← falso.
        Desapila pila_aristas. nodo ← Tope(pila_nodos).
    fin-si
  fin-si
fin-mientras
fin-si
fin-mientras
Retornar el valor de éxito.

```

Cada vez que se detecta un ciclo, es decir cuando se alcanza dos veces el mismo nodo en una dada pila, el siguiente paso es determinar si dicho ciclo contiene un hipergrafo parcial cuadrado  $\mathbf{PH} = (\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_n)$  donde  $\mathbf{N}_{\mathbf{PH}} = \{n_1, n_2, \dots, n_n\}$ . En ese caso, se ha detectado un subconjunto de asignación de orden  $n$ . Esta tarea es realizada por la subrutina *TestCiclo* presentada en el algoritmo 5.4.

Luego, se actualiza  $\mathbf{Haux}$  y el hipergrafo inducido cerrado  $\mathbf{H}_A$  de  $\mathbf{Haux}$  se asigna a  $\mathbf{Haux}$ , donde  $\mathbf{A} = \mathbf{Naux} - \mathbf{N}_{\mathbf{PH}}$ , mientras que el hipergrafo inducido cerrado  $\mathbf{H}_U$  de  $\mathbf{H}(\mathbf{R})$  se asigna a  $\mathbf{H}(\mathbf{R})$ , donde  $\mathbf{U} = \mathbf{N}_{\mathbf{H}(\mathbf{R})} \cup \mathbf{N}_{\mathbf{PH}}$ . En otras palabras, las  $n$  aristas y los  $n$  nodos de  $\mathbf{PH}$  se remueven de  $\mathbf{Haux}$  para ser incorporados a  $\mathbf{H}(\mathbf{R})$  de modo que  $\mathbf{A}_i$  se convierte en la  $(\text{ord}+i)$ -ésima arista de  $\mathbf{H}(\mathbf{R})$  y  $n_i$  en el  $(\text{ord}+i)$ -ésimo nodo de  $\mathbf{H}(\mathbf{R})$  para  $i=1, \dots, n$ .

Eventualmente, un bloque detectado puede corresponder a un subconjunto de asignación no permitido. Como se ha visto en el capítulo anterior, algunos subconjuntos de asignación pueden resultar inadmisibles debido a consideraciones físicas que van más allá de un mero análisis estructural. Los subconjuntos permitidos se almacenan en una lista denominada "Restricciones". La subrutina *ChequearNxN* se refiere a esta lista para verificar que el subconjunto asociado a  $\mathbf{PH}$  no esté incluido entre las restricciones. Si el subconjunto fuese inadmisibile, la subrutina continúa buscando ciclos sin modificar  $\mathbf{Haux}$  ni  $\mathbf{H}(\mathbf{R})$ .

**Algoritmo 5.4.** Función **TestCiclo**.

*Datos de entrada:* nodo\_ady, pila\_nodos, pila\_aristas, Restricciones

*Datos de entrada-salida:* **Haux**, **PH**,  $n$

*Valor retornado por la función:*

verdadero: si encuentra un subconjunto admisible,

falso: en caso contrario.

InicializarEnFalso(nodos).

RecuperarPH(nodo\_adj, pila\_nodos, pila\_aristas, **PH**).

edge  $\leftarrow$  1.

éxito  $\leftarrow$  falso.

aristasVisitadas  $\leftarrow$   $\emptyset$ .

**Mientras** queden aristas de **Haux** sin visitar **hacer**:

arista  $\leftarrow$  una arista de **Haux** no visitada.

aristasVisitadas  $\leftarrow$  aristasVisitadas  $\cup$  {arista}.

**Si** no Completo(arista, **Haux**, **HP**)

entonces RemoveArista(arista, **HP**).

fin-si

fin-mientras

nodosVisitados  $\leftarrow$   $\emptyset$ .

**Mientras** queden nodos de  $N_{\mathbf{Haux}}$  sin visitar **hacer**:

nodo  $\leftarrow$  un nodo de  $N_{\mathbf{Haux}}$  no visitado.

nodosVisitados  $\leftarrow$  nodosVisitados  $\cup$  {nodo}.

**Si** el nodo no pertenece a ninguna arista de **HP**

entonces RemoveNodo(nodo, **PH**).

fin-si

fin-mientras

**Si** Orden(**HP**) = a la cardinalidad de  $N_{\mathbf{HP}}$

entonces { **HP** es un hipergrafo parcial cuadrado de **Haux**}

**Si** ChequearNxN(**PH**,  $n$ , Restricciones)

entonces

éxito  $\leftarrow$  verdadero.

$n \leftarrow$  Orden(**HP**).

fin-si

fin-si

**Retornar** el valor de éxito.

CDHG termina cuando **Haux** está vacía ó cuando no se encuentran más subconjuntos de

asignación después de haber buscado empleando todas las reglas heurísticas. Es importante destacar que, al igual que con GS-FLCN, una vez que CDHG ha culminado la búsqueda, es necesario analizar los subconjuntos resultantes para determinar si todos ellos son aceptables. Si un subconjunto es rechazado, éste debe ser incorporado a Restricciones y el programa se corre nuevamente. Este procedimiento se repite hasta que se encuentra una solución satisfactoria. En este contexto, una solución es satisfactoria si no posee subconjuntos conflictivos.

### 5.2.2. HEURÍSTICAS

Las reglas heurísticas fueron diseñadas para determinar la mejor forma de moverse a través de **Haux**: dado un nodo  $n$  (nodo actual), cada regla elige entre sus nodos adyacentes el próximo que será visitado. El empleo de distintas reglas conduce a diferentes recorridos a través de **Haux**. Por esta razón, un ciclo que no ha sido detectado por una regla heurística puede ser alcanzado mediante otra. Es por eso que CDHG cambia de una regla a otra mediante la función **Próximo** y luego intenta de nuevo.

En este trabajo, se aplicaron las siguientes seis reglas heurísticas en el orden en que se listan. Los dos primeros criterios se basan en el grado de conectividad entre nodos adyacentes.

**Regla 1:** *Seleccione el nodo menos conectado.* Se elige el nodo con el mínimo grado de conectividad, es decir con la cantidad mínima de nodos adyacentes.

**Regla 2:** *Seleccione el nodo más conectado.* Se selecciona el nodo con el máximo grado de conectividad.

**Regla 3:** *Seleccione el nodo con menor etiqueta.* Se elige el nodo con la etiqueta más pequeña. Por ejemplo, si los nodos adyacentes tienen etiquetas 3, 15, 46 y 98, entonces se selecciona el que tiene etiqueta 3.

**Regla 4:** *Seleccione el nodo con mayor etiqueta.* Se selecciona el nodo con la mayor etiqueta. De acuerdo con este criterio, en el ejemplo anterior, el nodo 98 sería el próximo a visitar.

**Regla 5:** *Seleccione el nodo más cercano.* Para este criterio, definiremos distancia entre dos nodos como el valor absoluto de la diferencia entre sus etiquetas. La regla es elegir el nodo adyacente situado a la mínima distancia al nodo corriente. En el ejemplo mencionado arriba, si el nodo corriente tenía etiqueta 40, las respectivas distancias a los nodos 3, 15, 46 y 98 serían 37, 25, 6 y 58. Por lo tanto, se selecciona el nodo 46.

**Regla 6:** *Seleccione el nodo más alejado.* Se elige el nodo adyacente situado a la máxima distancia del nodo corriente. Para el ejemplo anterior, se seleccionaría el nodo 98.

Todas estas reglas heurísticas se diseñaron teniendo en cuenta la estructura típica de las matrices de ocurrencia asociadas a los modelos matemáticos de estado estacionario de las plantas de procesos químicos. La primera regla es apropiada para la detección de subconjuntos de asignación muy raros, donde es muy probable encontrar uno o más nodos débilmente conectados. Paralelamente, la regla 2 apunta a localizar subconjuntos con alta densidad de elementos no ceros. Tanto la tercera como la cuarta regla concentran la búsqueda en los extremos de la matriz de ocurrencia. La regla 5 tiende a efectuar búsquedas locales dentro de pequeñas regiones alrededor del nodo corriente. Esto resultó ser eficiente para diagramas de flujo con pocas corrientes de reciclo y/o siempre que los reciclos vinculan unidades vecinas. En contraste, la última regla es apropiada para plantas con reciclos que conectan unidades alejadas entre sí.

Además de emplear individualmente las heurísticas recién mencionadas, se aplicó también una regla de mezclado que permite la introducción de perturbaciones. La metodología se basa en los principios de "simulated annealing" debido a que consiste en seguir una regla heurística básica, desviándose de ella periódicamente en la dirección de otra regla. En particular, para el caso de estudio presentado aquí, se empleó una variante muy simple: se efectúa un paso de la regla del nodo más conectado cada dos pasos de la heurística del nodo menos conectado. En otras palabras, la Regla 1 es perturbada mediante la Regla 2.

Es importante mencionar que la calidad de las reglas heurísticas tiene influencia significativa en el nivel de las asignaciones que pueden lograrse. En este sentido, un conjunto de reglas que resulta apropiada para una dada familia de problemas puede resultar poco eficiente cuando se la emplea en otras aplicaciones. Por esta razón es fundamental analizar la

estructura de las matrices antes de efectuar la clasificación. Generalmente, las matrices ralas asociadas a la solución de problemas de una misma familia comparten ciertas características estructurales tales como densidad promedio, patrones de distribución de los elementos no nulos, simetrías, etc. (ver Duff y *co.* 1997). Por lo tanto, es aconsejable visualizar el patrón general de la matriz de ocurrencia para identificar su forma. Esto resulta útil para elegir apropiadamente las reglas heurísticas a emplear y la secuencia en que conviene aplicarlas.

Finalmente, se puede demostrar que los tiempos de ejecución de los algoritmos **BuscarAutoArista** y **BuscarHGP** son del orden  $O(n^2)$  y  $O(t)$  respectivamente, donde  $n$  es el orden de  $\mathbf{H}(\mathbf{N})$  y  $t$  es la cantidad de elementos no nulos en  $\mathbf{N}$ . Dado que el tiempo de ejecución de FLCN crece exponencialmente con el tamaño del subconjunto que estamos buscando, es evidente que CDHG es mejor en cuanto a eficiencia.

### 5.3. ANÁLISIS DEL DESEMPEÑO DE CDHG

CDHG fue comparado con GS-FLCN para varios casos de estudio con el objeto de analizar su desempeño. En esta tesis se presentan los resultados obtenidos para dos sectores de la planta industrial de producción de etano presentada en el capítulo anterior. El modelo empleado para llevar a cabo este análisis consistió de 90 unidades conectadas mediante 200 corrientes que involucran 12 componentes.

#### 5.3.1. RESULTADOS

La tabla 5.1 muestra los resultados de las corridas con ambos métodos. Es importante destacar que GS-FLCN se ejecutó con el máximo nivel posible de exploración, es decir sin factores de ramificación que podarían la búsqueda de subconjuntos de asignación.

Area	Compres-Criog.		Fraccionamiento	
Dimensión	772x529		333x258	
Tamaño	GS-FLCN	CDHG	GS-FLCN	CDHG
1	407	397	200	181
2	2	3	6	3
3	6	6	6	1
4	-	1	2	4
5	-	-	1	1
6	-	2	-	2
8	1	-	-	-
10	-	-	1	-
13	-	-	-	1
pva	82.6 %	82.6 %	98.4 %	91.5 %
Tiempo de ejecución (seg)	9450	68	1473	33

**Tabla 5.1.** Tiempos de ejecución de CDHG y GS-FLCN para la planta de etano.

La primera columna de la tabla indica el tamaño de los subconjuntos de asignación. Las últimas dos filas muestran el porcentaje de variables asignadas (*pva*) y los tiempos de ejecución, respectivamente. Los índices *pva* para las secciones criogénica y de compresión son los mismos para ambos métodos, pero el tiempo de ejecución requerido por CDHG es menor en varios órdenes de magnitud. CDHG también demostró mejor comportamiento para el área de fraccionamiento en cuanto a los tiempos de ejecución, aunque el *pva* disminuyó ligeramente.

Es importante puntualizar que el hecho de que las diferencias entre los *pvas* son pequeñas revela la buena calidad de las reglas heurísticas seleccionadas para guiar la búsqueda. En tal sentido, se apreció que el desempeño del método depende fuertemente de las heurísticas elegidas y/o diseñadas para el problema, dado que distintos tipos de problemas pueden derivar en diferentes patrones de realidad y niveles de densidad de elementos no nulos. Por tal motivo, no resulta factible proveer un conjunto de reglas heurísticas que sean efectivas para todos los posibles problemas a tratar. En definitiva, CDHG puede obtener la clasificación de variables en

menores tiempos de cómputo que GS-FLCN, pero GS-FLCN es significativamente más robusto e independiente del patrón de raridad de la matriz de ocurrencia.

## 5.4. ENFOQUE HÍBRIDO

Dadas las características propias de CDHG y GS-FLCN se decidió que la primera podría usarse exitosamente como una técnica de preprocesamiento previa a la aplicación de la segunda. De este modo, con CDHG se obtendría una primera solución razonable, con bajo costo computacional, identificando la mayoría de los subconjuntos de asignación. Luego, esta solución podría ser mejorada utilizando GS-FLCN para clasificar las variables no medidas que no pudieron ser asignadas por CDHG. Esto permitiría reducir el problema ingresado a GS-FLCN, lográndose de este manera mejores tiempos de ejecución globales. Esta nueva alternativa fue denominada *Enfoque Híbrido (EH)* (Ponzoni y co.,1998c).

### 5.4.1. ALGORITMO BÁSICO

Un esquema general de la misma se presenta en el algoritmo 5.5. En primer lugar se construye la matriz de ocurrencia  $\mathbf{N}$ , cuyas columnas corresponden a las variables no medidas del sistema de ecuaciones  $\mathbf{E}$ . Luego, CDHG es aplicado a  $\mathbf{N}$  con el fin de encontrar el mayor número posible de subconjuntos de asignación. Cuando no se pueden detectar nuevos subconjuntos de asignación con este método, las variables y ecuaciones aún no asignadas son agrupadas conformando una submatriz  $\mathbf{N}'$  de  $\mathbf{N}$ . Finalmente, se ejecuta GS-FLCN para buscar nuevos bloques en la submatriz  $\mathbf{N}'$ . Al finalizar la corrida de GS-FLCN, si se detectaron nuevos subconjuntos, estos son removidos de  $\mathbf{N}$  y se vuelve a ejecutar CDHG comenzando una nueva iteración del EH. Si GS-FLCN no logró hallar nuevos bloques, el EH termina. Es importante destacar que la submatriz  $\mathbf{N}'$  explorada por GS-FLCN es significativamente más pequeña que  $\mathbf{N}$

gracias al preprocesamiento efectuado por CDHG. De este modo, los tiempos de ejecución de GS-FLCN se reducen considerablemente.

---

**Algoritmo 5.5.** Enfoque Híbrido (**EH**).

1. Construir la matriz de ocurrencia **N** correspondiente a las variables no medidas.
  2. Aplicar CDHG a **N** hasta que no se encuentren nuevos subconjuntos de asignación.
  3. Construir la submatriz **N'** de **N** asociada con las variables y ecuaciones que no fueron asignadas por CDHG.
  4. Aplicar GS-FLCN a **N'** hasta que no se encuentren nuevos subconjuntos de asignación.
  5. **Si** GS-FLCN halló al menos un nuevo subconjunto de asignación **entonces**  
     Remover de M los subconjuntos detectados en M'.  
     Volver al paso 2.  
**si no** Terminar.
- 

#### 5.4.2. UNA APLICACIÓN INDUSTRIAL

A fin de establecer el desempeño del EH, se efectuó el análisis de observabilidad de la planta de producción de etano presentada en el capítulo anterior. El modelo matemático empleado para la planta en esta ocasión cuenta 1105 ecuaciones y 787 variables. Para CDHG se utilizaron las reglas heurísticas planteadas anteriormente. Por otra parte, GS-FLCN fue corrido hasta un nivel de profundidad 11, porque niveles de búsqueda más profundos requerían tiempos de cómputo excesivamente elevados. En este punto es importante destacar que los factores de ramificación (BF) no fueron empleados debido a que el objetivo de este estudio es comparar el GS-FLCN puro (sin podas en su espacio de búsqueda) frente al EH. En tal sentido, el EH y los BF son alternativas distintas para acelerar el GS-FLCN original. Ambas reducen el número de exploraciones. En el caso de los BF se acotan las ramas de la expansión forestal generada durante la exploración, mientras que con CDHG la búsquedas se concentran en distintas regiones de la matriz de ocurrencia según el conjunto de heurísticas diseñadas.

La tabla 5.2 resume la cantidad de conjuntos de asignación obtenidos con cada algoritmo. La primera columna indica el tamaño de los bloques mientras que la última refiere al porcentaje de variables calculado con respecto al total de variables no medidas.

De la tabla se aprecia que el EH exhibe un mejor desempeño dado que logra localizar mayor cantidad de conjuntos de asignación. Esto es claro en términos del *pva*. El alto valor de *pva* alcanzado por el EH se debe a que CDHG encuentra un subconjunto admisible de tamaño 13 que GS-FLCN no puede detectar dado que requiere explorar más allá del nivel 11. Por otra parte también se aprecia que EH determina más bloques de orden 1 y 2 que GS-FLCN, aún cuando ambas técnicas utilizan la misma política para detectar subconjuntos de estos tamaños. Esto obviamente se debe a que el desacople del subconjunto de orden 13 provocó el posterior desacople de bloques más pequeños.

Dimensión	1105x787	
Tamaño de bloques	Enfoque Híbrido	GS-FLCN
1	540	522
2	5	4
3	0	2
4	2	0
5	3	1
6	3	1
13	1	0
pva	76.75 %	69.50 %

**Tabla 5.2.** Conjuntos de asignación hallados por el Enfoque Híbrido y por GS-FLCN.

En general, se puede afirmar que el EH constituye una buena técnica para acelerar los tiempos de ejecución de GS-FLCN. No obstante, el éxito del EH depende en gran medida de la efectividad de CDHG, y por ende, depende indirectamente de la calidad de las heurísticas utilizadas por este algoritmo. Por lo tanto, esta técnica resulta aconsejable cuando se aplica a familias de problemas para los cuales se conoce el tipo de patrón estructural que siguen las matrices de ocurrencias, aplicando reglas heurísticas apropiadas para las mismas.

## 5.5. CONCLUSIONES

Se presentó en este capítulo un nuevo algoritmo basado en hipergrafos, denominado Detección de Ciclos en Hipergrafos (CDHG), que efectúa la clasificación de variables no medidas para problemas de instrumentación. CDHG es notoriamente superior en tiempos de ejecución a GS-FLCN, aunque no garantiza máxima robustez. De hecho, la calidad de las particiones logradas mediante CDHG depende fuertemente del diseño de las reglas heurísticas.

El consumo de tiempo requerido por el GS-FLCN puro se torna prohibitivo para problemas grandes. Por esta razón, también se propuso emplear CDHG a fin de obtener una descomposición básica de la matriz de ocurrencia, para luego aplicar GS-FLCN sobre el subproblema remanente (que será de menor tamaño) con el objeto de detectar mayor cantidad de subconjuntos de asignación formados por variables aún no asignadas. Esta estrategia fue denominada Enfoque Híbrido y demostró ser muy exitosa para el análisis de observabilidad de una planta de producción de etano de mediana dimensión.

La combinación de ambos algoritmos constituye una buena alternativa para lograr una partición satisfactoria de la matriz de ocurrencia sin costos computacionales exorbitantes. Si bien este enfoque preserva la dependencia de las heurísticas observada en CDHG, es posible afirmar que la mayoría de las matrices de ocurrencia asociadas a plantas de procesos poseen patrones de realidad similares con lo cual las heurísticas diseñadas resultan suficientes para el tratamiento de estos problemas. En líneas generales, sería posible diseñar nuevas reglas para explorar matrices con otras características y de este modo extender el rango de aplicación de la metodología. Así, cuando el patrón de realidad de las matrices de ocurrencia a procesar es conocido, el Enfoque Híbrido con heurísticas adecuadas resulta muy apropiado. Por el contrario, si no resulta factible predecir la estructura de las matrices, se recomienda emplear

exclusivamente GS-FLCN con factores de ramificación o bien el GS-pFLCN cuando se dispone de la infraestructura necesaria para realizar cómputo paralelo-distribuido.

# ***MÉTODO DIRECTO: OBSERVABILIDAD, BIGRAFOS Y DIGRAFOS***

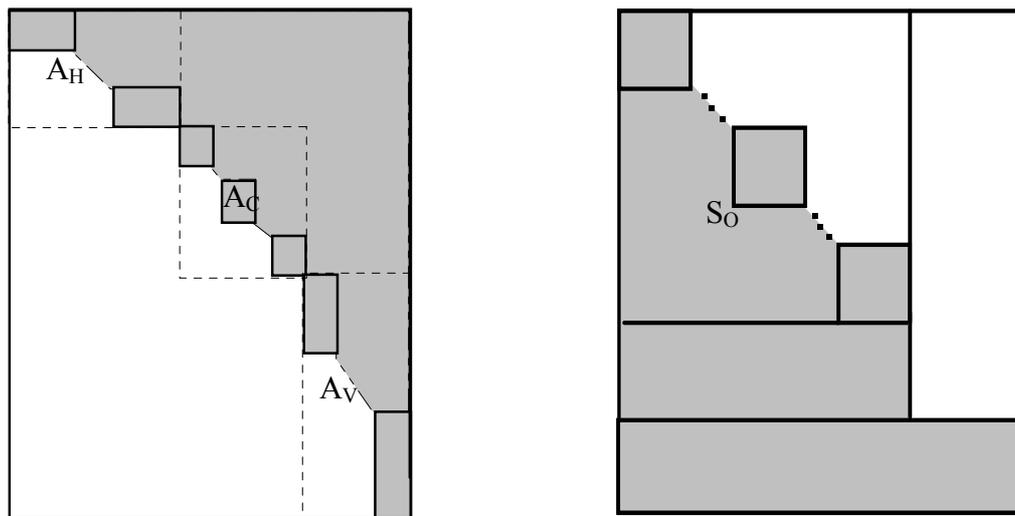
## **6.1. INTRODUCCIÓN**

En este capítulo se presenta un *método directo para análisis de observabilidad*. Esta técnica, basada en descomposición de grafos, constituye el último algoritmo de clasificación de variables desarrollado en el marco de esta tesis. La idea central surgió a partir del análisis de los métodos directos de reordenamiento de matrices existentes para la resolución de sistemas de ecuaciones lineales. Si bien el alcance y objetivo de estas técnicas resultaban completamente diferentes a los de nuestro problema, se consideró que la filosofía de reordenamiento estructural

de matrices contenida en estos algoritmos podía servir como base para el desarrollo de un método directo de clasificación de variables.

En la sección 2.5.2 de esta tesis se describió un método directo para permutar una matriz no cuadrada a FTB mediante la clásica descomposición de Dulmage-Mendelsohn. Básicamente, el procedimiento efectúa un reordenamiento en dos niveles. En la primer etapa, se asocia un grafo bipartito  $G$  a la matriz  $A$ , se halla un pareamiento maximal  $P$  de  $G$  y se clasifican los nodos de  $G$  en función de  $P$ . Luego, se reordenan las filas y columnas de  $A$  siguiendo la clasificación de nodos. Esta primera fase permuta  $A$  a la FTB mostrada en la figura 2.7. A continuación, en la segunda etapa, se particionan los tres bloques sobre la diagonal obteniéndose el patrón de la figura 6.1.a.

Observando la figura 6.1, resulta claro que el reordenamiento obtenido a partir de la descomposición de Dulmage-Mendelsohn difiere significativamente de la FTiB requerida por nuestra metodología estructural de análisis de observabilidad. El primero es triangular superior en bloques, donde únicamente los bloques correspondientes a  $A_C$  son cuadrados, mientras que los restantes ( $A_H$  y  $A_V$ ) son rectangulares. En cambio, el patrón requerido para observabilidad es triangular inferior en bloques, donde todos los bloques sobre la diagonal de la submatriz  $S_O$  son cuadrados. Es evidente que no es posible pasar de una FTB a la otra mediante una simple operación de transposición de matrices. En este capítulo, se describe una metodología que permite aprovechar los conceptos en que se basa la técnica de Dulmage y Mendelsohn para obtener la FTiB para análisis de observabilidad en forma directa.



a) FTB de Dumalge y Mendelsohn

b) FTiB para análisis de observabilidad

*Figura 6.1. Comparación de FTBs.*

## 6.2. MÉTODO DIRECTO PARA ANÁLISIS DE OBSERVABILIDAD

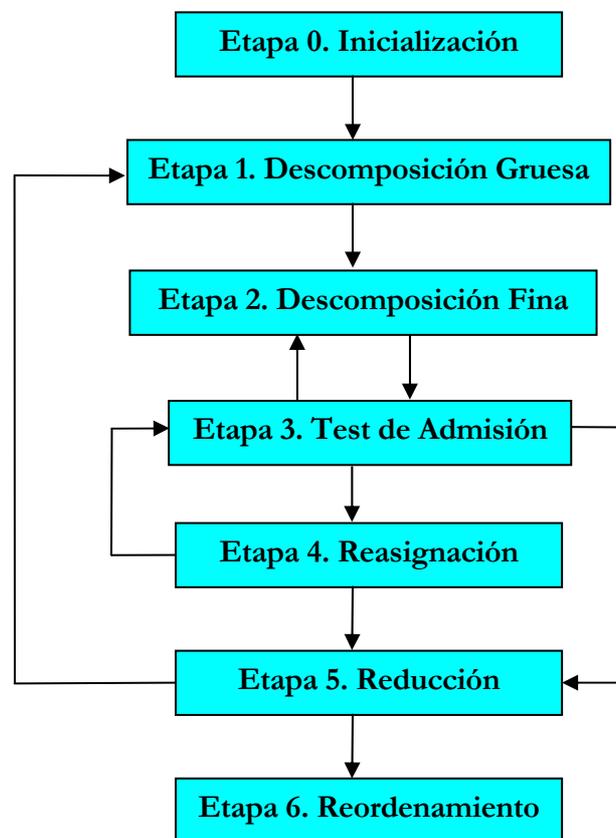
A pesar de que la descomposición de Dulmage-Mendelsohn no resultaba apropiada para nuestro problema, esta metodología sirvió de inspiración para el desarrollo del método directo que describiremos a continuación.

### 6.2.1. ESTRUCTURA GENERAL DEL MÉTODO

El algoritmo diseñado por nosotros consta de seis etapas, más una fase de inicialización, tal como muestra el esquema de la figura 6.2. En primer lugar se construye el bigrafo  $G$  asociado a la submatriz de ocurrencia  $N$ . Después se efectúa la descomposición gruesa de  $G$  mediante la búsqueda de un pareamiento maximal de  $G$ . En la etapa 2, se efectúa un segundo nivel de particionamiento en donde se determinan los subconjuntos de asignación de  $N$ . Cada uno de estos subconjuntos es testeado en la etapa 3. Cuando un bloque resulta no admisible, se

intenta efectuar una reasignación (etapa 4); este proceso trata de sustituir una de las filas del subconjunto rechazado por alguna fila correspondiente a una ecuación redundante.

Si la reasignación es exitosa, el algoritmo vuelve al paso 2; en caso contrario se pasa a la etapa 5 de reducción. En esta fase, se remueven de  $G$  todos los nodos asociados a las filas y columnas de los bloques que superaron el test de admisión. Luego, con el bigrafo reducido, se regresa a la etapa 1. Cuando ya no se detectan nuevos subconjuntos de asignación, el algoritmo pasa a la etapa 6 donde se produce el reordenamiento y termina el procedimiento.



*Figura 6.2. Esquema general del método directo*

Dada esta descripción básica de la metodología, vamos a concentrarnos ahora en explicar en detalle cada una de las etapas del método. En el algoritmo 6.1 se resumen los pasos del método directo.

En la inicialización se construye el grafo bipartito  $G = (R, C, A)$  asociado a la submatriz de ocurrencia  $N$  tal como se mostró en el capítulo 2; es decir, los nodos en  $R$  y  $C$  están vinculados a las filas y columnas de  $N$  respectivamente, y las aristas de  $A$  representan los elementos no nulos de la matriz.

**Algoritmo 6.1.** Método Directo para Clasificación de Variables.

D.E.:  $N$  (matriz de ocurrencia) y  $R$  (restricciones iniciales)

D.S.:  $N$  reordenada a FTiB.

**Etapa 0. Inicialización.**

0.1 Construir el bigrafo  $G(N) = (R, C, E)$  asociado a  $N$ .

**Etapa 1. Descomposición Gruesa.**

1.1 Obtener un pareamiento maximal  $P_m$  de  $G(N)$ .

1.2 Clasificar en base a  $P_m$  las filas en  $SR1$ ,  $SR2$ ,  $VR$ ,  $HR$  y las columnas en  $SC1$ ,  $SC2$  y  $HC$ .

1.3 Para cada *fila especial* que exista, clasificarla como:

**VR**, si todas sus aristas conducen a columnas de **SC1** ó **SC2**.

**SR1**, si está conectada a una única columna de **HC** y a ninguna columna de **SC2** (en este caso dicha columna y la fila especial conforman un bloque de  $1 \times 1$ ).

**SR2**, si está conectada a una única columna de **HC** y al menos a una columna de **SC2** (en este caso dicha columna y la fila especial forman un bloque de  $1 \times 1$ ).

**HR**, si está conectada a más de una columna de **HC**.

**Etapa 2. Descomposición Fina.**

2.1 Asociar el digrafo  $G(N_1) = (V, E)$  a la matriz  $N_1$  correspondiente al bloque  $(SR1, SC1)$ .

2.2 Descomponer  $G(N_1)$  en sus componentes fuertes  $N_{11}$ ,  $N_{12}$ , ...,  $N_{1q}$ . Cada  $N_{1i}$  corresponde a un bloque de la diagonal.

2.3 Asociar el digrafo  $G(N_2) = (V, E)$  a la matriz  $N_2$  correspondiente al bloque  $(SR2, SC2)$ .

2.4 Descomponer  $G(N_2)$  en sus componentes fuertes  $N_{21}$ ,  $N_{22}$ , ...,  $N_{2p}$ . Cada  $N_{2i}$  corresponde a un bloque de la diagonal.

**Etapa 3. Test de Admisión.**

3.1 Para cada componente fuerte  $N_{1i}$ ,

3.1.1 Chequear que  $N_{1i}$  no esté en el conjunto de restricciones  $R$ .

3.1.2 Si  $N_{1i}$  está prohibido, ir a la Etapa 4 de Reasignación.

3.1.3 Si la reasignación tuvo éxito, volver a la etapa 2. En caso contrario, ir a la Etapa 5 de Reducción  $N_{1i}$ .

3.2 Para cada componente fuerte  $N_{2i}$

3.2.1 Chequear que  $N_{2i}$  no esté

en el conjunto de restricciones  $R$ .

- 3.2.2 Si  $N_{2i}$  está prohibido,  
ir a la Etapa 5 de Reducción con  $N_{2i}$ .

3.3 Ir a la Etapa 6.

**Etapa 4. Reasignación de una componente fuerte  $N_{1i}$ .**

4.1 Buscar una arista  $(r,c)$ , donde  $r \in VR$ ,  $c$  pertenece a las columnas presentes en  $N_{1i}$ ,  $(k,c) \in P_m$  y  $k$  no fue reasignada por  $r$  anteriormente.

4.2 Si tal arista existe, entonces la reasignación es posible y se deben llevar a cabo las siguientes acciones:

4.2.1 Eliminar de  $P_m$  la arista  $(k,c)$

donde  $k$  es una de las filas de  $N_{1i}$ .

4.2.2 Agregar  $(r,c)$  a  $P_m$ ; eliminar  $k$  de  $SR_1$  y  $N_{1i}$ .

4.2.3 Agregar  $r$  a  $SR_1$  y  $M_{1i}$ ; eliminar  $r$  de  $VR$ .

4.2.4 Agregar  $k$  a  $VR$ .

En caso contrario, la reasignación no es posible.

4.3 Volver al paso 3.1.3.

**Etapa 5. Reducción del bigrafo  $G(M)$  hasta la componente fuerte  $N_{ij}$ .**

5.1 Eliminar del bigrafo todas las filas y columnas correspondientes a las componentes fuertes anteriores a  $N_{ij}$  - es decir, la  $N_{kl}$  con  $k < i$  ó  $k = i$  y  $l < j$  - e incorporarlas a la solución.

5.2 Seleccionar una fila de  $N_{ij}$  como *fila especial* (que no haya sido seleccionada anteriormente) y eliminarla de  $G(N)$ .

5.3 Si todas las filas de  $N_{ij}$  fueron seleccionadas anteriormente como *fila especial*, seleccionar dos *filas especiales* entre las filas de  $N_{ij}$ , sin importar si haya habían sido elegidas antes.

5.4 Volver a la etapa 1.

**Etapa 6. Reordenamiento.**

6.1 Reordenar  $N$  como sigue:

$[N_{11}, N_{12}, \dots, N_{1p}, N_{21}, N_{22}, \dots, N_{2q}, (VR, SC_1), (HR, HC)]$

6.2 Fin del Algoritmo.

**6.2.2. DESCOMPOSICIÓN GRUESA**

Una vez generado  $G$  se efectúa el primer nivel de particionamiento, al cual hemos denominado *Descomposición Gruesa*. En esta etapa se busca un pareamiento maximal  $P_m$  de  $G$  empleando el algoritmo *PMB* presentado en la sección 2.4. Luego, se procede a clasificar los nodos en función de  $P_m$ .

Como se mencionó en la introducción de este capítulo, la forma en que Dulmage y Mendelsohn clasifican los nodos de  $G$  no conduce al patrón de reordenamiento deseado. Por tal

motivo, en esta tesis se propone una nueva forma de categorizar los nodos de  $G$  (Ponzoni y *co.*, 1998b) que particiona los conjuntos de nodos,  $R$  y  $C$ , en los siguientes subconjuntos disjuntos:

$$V_R = \{\text{nodos no apareados de } R\}$$

$$S_{R1} = \{\text{nodos apareados de } R \text{ alcanzables desde algún nodo no apareado de } R \text{ mediante un camino alternante}\}$$

$$H_R = \{\text{nodos apareados de } R \text{ alcanzables desde algún nodo no apareado de } C \text{ mediante un camino alternante}\}$$

$$S_{R2} = R \setminus (V_R \cup S_{R1} \cup H_R)$$

$$S_{C1} = \{\text{nodos apareados de } C \text{ alcanzables desde algún nodo no apareado de } R \text{ mediante un camino alternante}\}$$

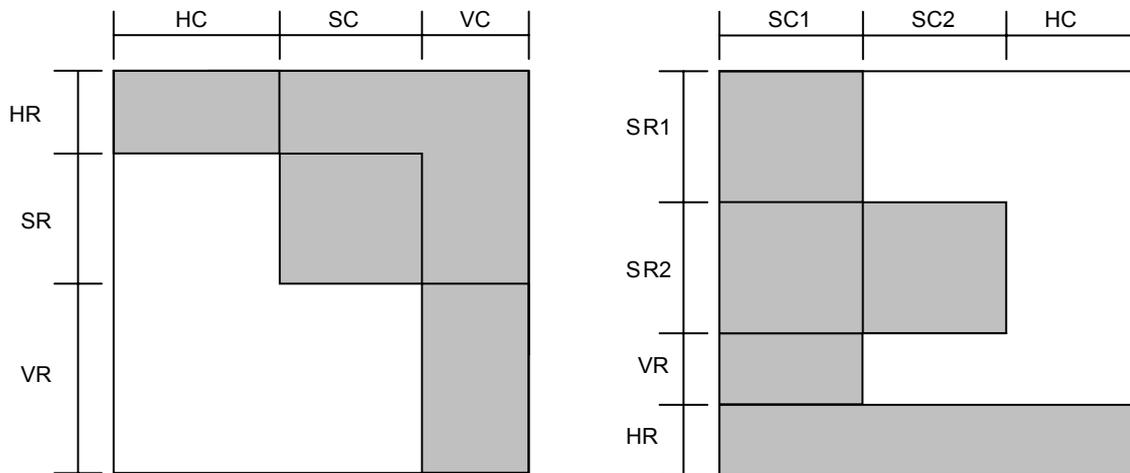
$$H_C = \{\text{nodos de } C \text{ tanto apareados como no apareados alcanzables desde algún nodo no apareado de } C \text{ mediante un camino alternante}\}$$

$$S_{C2} = C \setminus (S_{C1} \cup H_C)$$

La principal diferencia con la descomposición de Dulmage y Mendelsohn presentada en el capítulo 2 consiste en que ésta agrupa los nodos de  $R$  pertenecientes a  $V_R$  y  $S_{R1}$  en un único conjunto, mientras que nuestra clasificación requiere distinguirlos. En la figura 6.3 se aprecia el contraste entre ambas clasificaciones. En la figura 6.3.b se muestra además la forma en que los nodos deben ser reordenados para obtener nuestra FTiB. Nótese que los bloques  $(S_{C1}, S_{R1})$  y  $(S_{C2}, S_{R2})$ , además de ser cuadrados, tienen transversal llena dado que los nodos de  $S_{C1}, S_{R1}, S_{C2}$  y  $S_{R2}$  pertenecen al pareamiento maximal.

En la figura 6.4.a se presenta un ejemplo de nuestra partición para la matriz  $N$  de la figura 6.4.b. Luego, en la figura 6.4.c se muestra la matriz  $N$  reordenada de acuerdo con nuestra descomposición gruesa. Se puede observar que los nodos de  $S_{C1}$  y  $S_{C2}$  están asociados a las variables observables, mientras que los de  $H_C$  corresponden a las variables indeterminables. Asimismo, los nodos de  $S_{R1}$  y  $S_{R2}$  representan las ecuaciones asignadas, los nodos de  $V_R$

representan las ecuaciones redundantes y los de HR corresponden a las ecuaciones con variables indeterminables.



a) Clasificación de Dulmage-Mendelsohn.

b) Nueva clasificación de nodos.

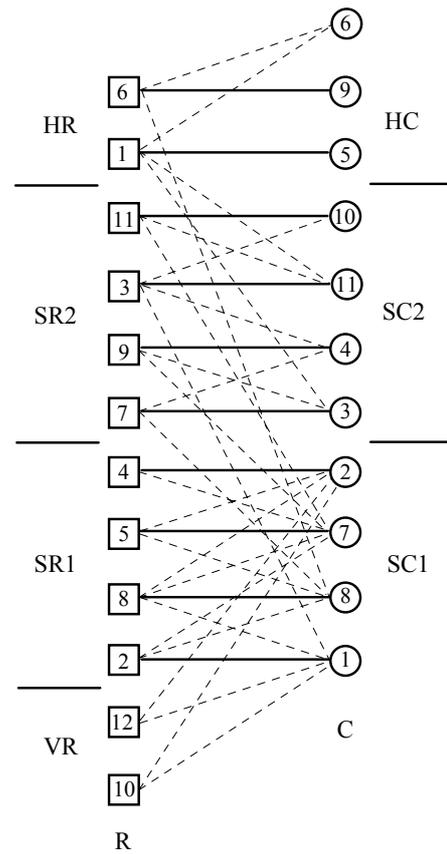
**Figura 6.3.** Clasificación de nodos y descomposición gruesa

### 6.2.3. DESCOMPOSICIÓN FINA

La descomposición gruesa obtiene dos bloques cuadrados estructuralmente no singulares determinados por los conjuntos  $(SR1, SC1)$  y  $(SR2, SC2)$ . La etapa de *descomposición fina* particiona estos bloques en subsistemas cuadrados no reducibles mediante el algoritmo de detección de las componentes fuertes de un digrafo (TCCD) presentado en el capítulo 2. En primer término se descompone  $G(N_1)$ , donde  $N_1 = (SR1, SC1)$ ; y luego  $G(N_2)$ , donde  $N_2 = (SR2, SC2)$ . Cada una de las componentes fuertes detectadas en estos digrafos corresponde a un subconjunto de asignación de la FTiB.

$$N = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

a) Matriz de ocurrencia



b) Pareamiento Maximal de  $G(N)$ .

	②	⑦	⑧	①	⑩	⑪	④	③	⑥	⑨	⑤
④	1	1	0	0	0	0	0	0	0	0	0
⑤	1	1	1	0	0	0	0	0	0	0	0
⑧	1	1	1	1	0	0	0	0	0	0	0
②	0	1	1	1	0	0	0	0	0	0	0
⑪	0	1	0	0	1	1	0	0	0	0	0
③	0	0	0	1	1	1	1	0	0	0	0
⑨	0	1	0	0	0	0	1	1	0	0	0
⑦	0	0	1	0	0	0	1	1	0	0	0
⑫	1	0	0	1	0	0	0	0	0	0	0
⑩	1	0	0	1	0	0	0	0	0	0	0
⑥	0	0	1	0	0	0	0	0	1	1	0
①	0	0	0	0	0	1	0	1	1	0	1

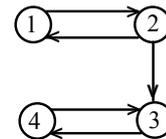
c) Matriz reordenada a partir de la descomposición gruesa

Figura 6.4. Descomposición Gruesa.

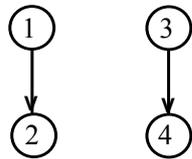
En la figura 6.5 se muestra la descomposición fina del bloque  $N_2 = (SR_2, SC_2)$  correspondiente al ejemplo de la figura 6.4. En primer lugar se construye el digrafo  $G(N_2)$  correspondiente al bloque. Esto se observa en las figuras 6.4.a y 6.4.b. Luego se particiona el digrafo en sus componentes fuertes (figura 6.4.c). Finalmente se reordena  $N_2$  de acuerdo a las componentes fuertes halladas por TCCD.

	1	2	3	4
1	1	1	0	0
2	1	1	1	0
3	0	0	1	1
4	0	0	1	1

a) Bloque  $N_2 = (SC_2, SR_2)$ .



b) Componentes fuertes de  $G(N_2)$ .



c) Componentes fuertes de  $G(N_2)$ .

	3	4	1	2
3	1	1	0	0
4	1	1	0	0
1	0	0	1	1
2	1	0	1	1

d) Bloque  $N_2$  reordenado.

**Figura 6.5.** Descomposición Fina de  $N_2 = (SC_2, SR_2)$ .

#### 6.2.4. TEST DE ADMISIÓN, REASIGNACIÓN DE BLOQUES Y REDUCCIÓN DEL BIGRAFO

El siguiente paso del algoritmo consiste en verificar la admisibilidad de los subconjuntos de asignación encontrados en la descomposición fina. El test de admisión empleado es el mismo que se utilizó en GS-FLCN (ver sección 4.4.4); es decir, se chequea cada bloque a fin de asegurar que no contenga alguno de los subconjuntos prohibidos especificados en el conjunto de restricciones  $R$ , donde  $R$  almacena información sobre subsistemas de  $N$  que por alguna razón no resultan admisibles.

En primer lugar se analizan los bloques  $N_{1i}$ , con  $i = 1..p$ , correspondientes a la submatriz  $N_1 = (SR_1, SC_1)$ . Si un bloque  $N_{1j}$ , con  $1 \leq j \leq p$ , contiene un subconjunto prohibido  $T$  de  $R$ , el algoritmo pasa a la etapa de reasignación. En esta fase, se busca un nodo fila  $r$  de  $VR$  tal que

exista una arista  $(r, c)$ , siendo  $c$  un nodo columna de  $\mathbf{N}_{1j}$ . Si dicho nodo existe, el método reemplaza el nodo fila  $k$  de  $\mathbf{N}_{1j}$  que estaba apareado a  $c$  por el nodo  $r$ . En términos del sistema de ecuaciones, la reasignación intercambia una de las ecuaciones del subsistema que falló en el test de admisión por una de las ecuaciones redundantes. Si se observa la figura 6.3.b. se puede apreciar que la reasignación efectúa una permutación entre filas de SR1 y VR.

Una vez efectuada esta permutación, el digrafo correspondiente a  $\mathbf{N}_1$  se construye nuevamente y se vuelve a efectuar la descomposición fina del bloque. El objetivo de esta estrategia de reasignación (propuesta en Ponzoni y co., 1998d) es romper la estructura del bloque  $\mathbf{N}_{1j}$  que contenía al subconjunto prohibido  $\mathbf{T}$ . Al cambiar un nodo fila de  $\mathbf{N}_{1j}$  por un nodo de VR, el bloque rechazado  $\mathbf{N}_{1j}$  no se volverá a formar en la siguiente descomposición fina de  $\mathbf{N}_1$ .

Cuando no es posible reasignar el bloque  $\mathbf{N}_{1j}$ , el método ejecuta la etapa de reducción del bigrafo. En esta fase, los bloques  $\mathbf{N}_{1i}$ , con  $i < j$ , son almacenados en la estructura que mantiene los subconjuntos de asignación válidos. Luego, los nodos fila y columna de dichos bloques son removidos del bigrafo  $\mathbf{G}(\mathbf{N})$ . En este punto, se selecciona un nodo fila  $e$  de  $\mathbf{N}_{1j}$ , a la cual denominaremos *fila especial*, y se la remueve temporalmente de  $\mathbf{G}(\mathbf{N})$ . Luego, el control se transfiere nuevamente a la primera etapa del método.

La selección y remoción de una fila especial responde a la necesidad de hallar un pareamiento maximal  $P_m'$  diferente al obtenido en la primera ejecución de la descomposición gruesa. Si no se removiera temporalmente algún nodo fila de  $\mathbf{G}(\mathbf{N})$ , la siguiente ejecución del paso 1 del algoritmo construiría un pareamiento maximal  $P_m'$  equivalente al  $P_m$  encontrado originalmente. La única diferencia sería que  $P_m'$  no contendría los nodos de los bloques  $\mathbf{N}_{1i}$  eliminados de  $\mathbf{G}(\mathbf{N})$ , pero el resto de  $P_m$  sería igual a  $P_m'$ .

Luego de generado el nuevo pareamiento maximal  $P_m'$ , la fila especial  $e$  es restituida al bigrafo y clasificada según las siguientes reglas:

- $e$  pertenecerá a VR, si todas sus aristas conducen a columnas de SC1 ó SC2.
- $e$  pertenecerá a SR1, si  $e$  está conectado a una única columna de HC y a ninguna columna de SC2 (en este caso dicha columna y  $e$  conforman un bloque de orden 1).
- $e$  pertenecerá a SR2, si  $e$  está conectado a una única columna de HC y al menos a una columna de SC2 (en este caso dicha columna y  $e$  forman un bloque de orden 1).
- $e$  pertenecerá a HR, si  $e$  está conectada a más de una columna de HC.

**Teorema 6.1.** Dado que el nodo  $e$  pertenecía al bloque rechazado  $N_{1j}$ , el bloque  $N_{1j}$  no se volverá a formar en la siguiente ejecución de la descomposición fina.

Demostración: la justificación es por el absurdo. Téngase en cuenta en primer lugar que las restricciones siempre corresponden a subsistemas de orden 2 o mayor. Luego, si  $N_{1j}$  fue rechazado en el test de admisión,  $N_{1j}$  tiene que ser necesariamente de dimensión superior a 1 porque contiene una restricción. Pero por otra parte, las reglas de clasificación enunciadas arriba establecen que  $e$  sólo puede pertenecer a un bloque de orden 1. Luego,  $N_{1j}$  no estará presente en la nueva descomposición fina.

De este modo, la nueva clasificación de nodos conduce a un nueva descomposición de  $G(N)$  sin perder nodos, dado que la fila especial  $e$ , eliminada temporalmente, es reestablecida luego de obtenerse el nuevo pareamiento maximal. Esto permite dejar atrás los bloques que no pasan el test mediante la construcción de diferentes descomposiciones.

Una vez que todos los bloques de  $N_1$  fueron chequeados exitosamente, el test de admisión se efectúa sobre el bloque  $N_2 = (SR2, SC2)$ . A diferencia con  $N_1$ , los bloques rechazados de

$N_2$  no pueden ser reasignados. La razón resulta obvia si se observa nuevamente la figura 6.3.b. Examinando el patrón de la FTiB resulta claro que las filas redundantes sólo poseen elementos no nulos en las columnas correspondientes a  $SR1$ . Luego, la reasignación carece de sentido para  $N_2$ . Por tal motivo, cuando un bloque  $N_{2i}$  de  $N_2$  no es admisible, el algoritmo pasa directamente a la etapa de reducción del bigrafo.

En algunas situaciones, puede suceder que la remoción de una única fila especial no resulte suficiente para encontrar nuevos bloques admisibles. En estos casos, el método elimina varias filas especiales hasta lograr un avance en la descomposición; es decir, distintos nodos filas son eliminados temporalmente del bigrafo hasta lograr el desacople de nuevos bloques. La política de clasificación de los nodos borrados transitoriamente es siempre la misma. Una vez que se encuentra el nuevo pareamiento maximal, cada una de las filas especiales es reincorporada y clasificada siguiendo las reglas enunciadas anteriormente. El algoritmo termina cuando todos los bloques de  $N_2$  han sido aceptados por el test de admisión o cuando los conjuntos  $SR1$  y  $SR2$  retornados por la descomposición gruesa están vacíos.

A modo de ilustración, considérese nuevamente el ejemplo de la figura 6.4. La descomposición fina del bloque  $N_1 = (SR1, SC1)$  obtiene una única componente fuerte, la cual obviamente está conformada por todas las filas y columnas de  $N_1$ . Supongamos que existe en  $R$  una restricción  $T$  formadas por los nodos fila  $\{2, 4, 5, 8\}$  y los nodos columna  $\{2, 7, 8, 1\}$ , dado que las ecuaciones y variables asociadas a dichas filas y columnas corresponden a un subsistema no singular del sistema completo de ecuaciones algebraicas  $E$  que modela la planta. Luego, el bloque es rechazado por el test de admisión y se pasa a la etapa de reasignación. Asumamos ahora que se seleccionan e intercambian los nodos fila 2 de  $SR1$  y 10 de  $VR$ . El bloque  $N_1$  pasa a estar formado ahora por los nodos fila  $\{4, 5, 8, 10\}$  tal como muestra la figura 6.6.a, mientras que el nodo 2 corresponde a una ecuación redundante.

	②	⑦	⑧	①	⑩	⑪	④	③	⑥	⑨	⑤
④	1	1	0	0	0	0	0	0	0	0	0
⑤	1	1	1	0	0	0	0	0	0	0	0
⑧	1	1	1	1	0	0	0	0	0	0	0
⑩	1	0	0	1	0	0	0	0	0	0	0
⑪	0	1	0	0	1	1	0	0	0	0	0
③	0	0	0	1	1	1	1	0	0	0	0
⑨	0	1	0	0	0	0	1	1	0	0	0
⑦	0	0	1	0	0	0	1	1	0	0	0
⑫	1	0	0	1	0	0	0	0	0	0	0
②	0	1	1	1	0	0	0	0	0	0	0
⑥	0	0	1	0	0	0	0	0	1	1	0
①	0	0	0	0	0	1	0	1	1	0	1

a) Matriz después de la primer reasignación.

	①	②	⑦	⑧	⑩	⑪	④	③	⑥	⑨	⑤
⑩	1	1	0	0	0	0	0	0	0	0	0
⑫	1	1	0	0	0	0	0	0	0	0	0
⑤	0	1	1	1	0	0	0	0	0	0	0
⑧	1	1	1	1	0	0	0	0	0	0	0
⑪	0	0	1	0	1	1	0	0	0	0	0
③	1	0	0	0	1	1	1	0	0	0	0
⑨	0	0	1	0	0	0	1	1	0	0	0
⑦	0	0	0	1	0	0	1	1	0	0	0
④	0	1	1	0	0	0	0	0	0	0	0
②	1	0	1	1	0	0	0	0	0	0	0
⑥	0	0	0	1	0	0	0	0	1	1	0
①	0	0	0	0	0	1	0	1	1	0	1

b) Matriz después de la segunda reasignación.

**Figura 6.6.** Reasignación del bloque  $N_1$ .

Supongamos que la descomposición fina vuelve a ejecutarse encontrándose nuevamente una única componente fuerte integrada por todo el bloque  $N_1$ . Asumamos ahora que el nuevo bloque, formado por los nodos filas  $\{4, 5, 8, 10\}$  y los nodos columna  $\{2, 7, 8, 1\}$ , es chequeado y rechazado debido a que contiene una restricción formada por los nodos fila  $\{4, 5, 10\}$  y los nodos columna  $\{2, 7, 8\}$ . Esto provoca una nueva reasignación, en la cual supongamos que se

intercambian los nodos fila 4 y 12 quedando el bloque  $\mathbf{N}_1$  compuesto por los nodos filas {5, 8, 10, 12} y los nodos columna {2, 7, 8, 1}. Se repite la ejecución de la descomposición fina, obteniéndose ahora dos componentes fuertes  $\mathbf{N}_{11} = \{\{10,12\}, \{1,2\}\}$  y  $\mathbf{N}_{12} = \{\{5,8\}, \{7,8\}\}$ . Ambos superan el chequeo, y el test de admisión finaliza para  $\mathbf{N}_1$ . La matriz resultante se muestra en la figura 6.6.b.

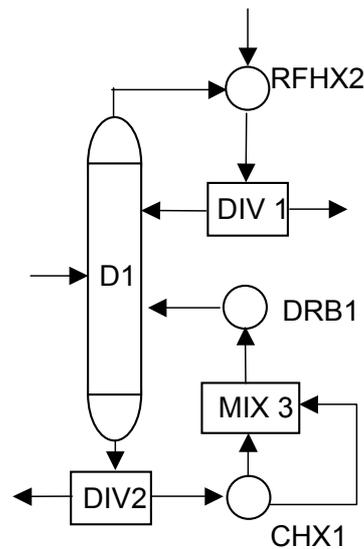
### 6.3. ANÁLISIS DE DESEMPEÑO DEL MÉTODO DIRECTO

El método directo fue empleado para resolver la clasificación de variables no medidas en varios problemas reales. Los resultados alcanzados demostraron que esta técnica resulta muy robusta y eficiente en tiempos de cómputo. A fin de efectuar una comparación de desempeño entre el método directo y GS-FLCN, en esta sección se presentan tres casos de estudio para los cuales se efectuó el análisis de observabilidad empleando ambos algoritmos.

#### 6.3.1. CASO DE ESTUDIO I: COLUMNA DE DESTILACIÓN.

El primer ejemplo corresponde a la sección de purificación de amoníaco por destilación de una planta industrial de síntesis de amoníaco. Este sector, cuyo diagrama de flujo se presenta en la figura 6.7, posee una columna de destilación (D1), un mezclador (MIX3), dos divisores (DIV1 y DIV2), y tres intercambiadores de calor (CHX1, DRB1 y RFHX2). El modelo matemático del sector está formado por 104 ecuaciones algebraicas no lineales con 25 variables medidas y 85 variables no medidas.

Los resultados de los análisis de observabilidad efectuados con ambos métodos se esquematizan en la tabla 6.1. Es importante mencionar que, al tratarse de un problema de pequeña dimensión, GS-FLCN pudo ser ejecutado sin necesidad de emplear factores de ramificación, razón por lo cual el espacio de búsqueda fue explorado en su totalidad.



**Figura 6.7.** Diagrama de flujo de la columna de destilación.

Dimensión	104x100	
	GS-FLCN	Método directo
Tamaño de bloques		
1	55	44
2	1	1
5	-	1
6	1	-
12	-	1
pva	74%	74%
tiempo de ejecución (en minutos)	1:53	0:03

**Tabla 6.1.** Resultados del análisis de observabilidad para el caso de estudio I.

Ambos métodos lograron determinar las 63 variables observables que posee el problema, por lo cual el porcentaje de variables asignadas (*pva*) es el mismo para los dos algoritmos. Esto se aprecia en la anteúltima fila de la tabla. No obstante esto, la conformación de los subconjuntos de asignación es distinta en cada caso.

En general se aprecia que GS-FLCN obtiene bloques más pequeños que el método directo. Esto se debe a la forma en que este algoritmo busca los subconjuntos de asignación. GS-FLCN sólo inicia la detección de bloques de tamaño  $n$  cuando se agotó totalmente la exploración de subconjuntos de dimensión  $n-1$ . Por otra parte, cada vez que el método detecta

un bloque de tamaño  $n$ , con  $n > 1$ , lo desacopla y reinicia la exploración para detectar subconjuntos de orden 1 nuevamente. Esta política asegura conjuntos de mínimo tamaño, siempre que no se utilicen factores de ramificación o cualquier otra técnica que acote el espacio de exploración. Esta es una cualidad importante de GS-FLCN dado que los bloques más pequeños facilitan la detección de subconjuntos de asignación no válidos y la incorporación de restricciones. En cuanto a los tiempos de ejecución se observa un desempeño muy superior del método directo.

Los resultados alcanzados muestran que, para este ejemplo, GS-FLCN logra mayor robustez (detección de bloques de mínimo tamaño) que el método directo, aún cuando ambos logran el mismo nivel de eficacia (cantidad de variables observables detectadas). Como conclusión, se puede afirmar que cuando GS-FLCN es empleado sin técnicas que poden el espacio de búsqueda, el algoritmo logra una descomposición de granularidad igual o más fina que el método directo. No obstante, cabe destacar que para plantas industriales grandes los modelos matemáticos suelen tornarse más complejos y grandes, razón por lo cual no siempre resultará factible utilizar GS-FLCN en forma pura, sin factores de ramificación u otras estrategias de aceleración.

### **6.3.2. CASO DE ESTUDIO II: PLANTA DE SÍNTESIS DE AMONÍACO.**

El segundo ejemplo corresponde a la planta de síntesis de amoníaco descrita en la sección 4.5.2.1. El modelo matemático fue generado nuevamente, con algunas opciones termodinámicas diferentes a las empleadas en el capítulo 4, quedando integrado en esta oportunidad por 557 ecuaciones y 587 variables. Se configuraron pocos instrumentos sobre la topología de la planta, razón por la cual sólo 74 variables quedaron medidas y las 513 restantes como variables no medidas.

Para este caso de estudio, de tamaño mediano respecto al ejemplo anterior, fue necesario el uso parcial de factores de ramificación al ejecutar GS-FLCN para evitar tiempos de cómputo excesivamente altos. Dado que los tiempos crecen a medida que aumenta la profundidad de búsqueda, para los niveles de exploración correspondientes a los bloques más pequeños (de tamaño 6 o menor) no se redujo el espacio de búsqueda, ejecutándose GS-FLCN en forma pura. A partir del nivel 7 los factores de ramificación fueron reduciendo gradualmente dicho espacio, alcanzándose las cotas más estrictas para los niveles 13 y posteriores.

En la tabla 6.2 se presentan los resultados obtenidos por GS-FLCN y el método directo. A diferencia con el caso de estudio anterior, aquí se aprecia que el método directo logra un porcentaje de variables asignadas (*pva*) ligeramente superior al alcanzado por GS-FLCN. Todas las variables asignadas por GS-FLCN fueron también clasificadas como observables por el método directo. Sin embargo, 34 variables observables detectadas por el método directo no pudieron ser asignadas por GS-FLCN.

La causa para esta diferencia entre los conjuntos de variables asignadas por cada método radica en que GS-FLCN no puede explorar el espacio de búsqueda en toda su extensión para subconjuntos de tamaño 7 en adelante. Como consecuencia de la poda, algunos subconjuntos de asignación válidos no son detectados por el algoritmo.

<b>Dimensión</b>	<b>557x513</b>	
<b>Tamaño de bloques</b>	<b>GS-FLCN</b>	<b>Método directo</b>
1	113	117
2	2	-
5	-	2
6	2	2
7	-	1
8	-	1
9	-	1
pva	25%	32%
tiempo de ejecución		

(en minutos)	35:12	0:14
--------------	-------	------

**Tabla 6.2.** Resultados del análisis de observabilidad para el caso de estudio II.

Como hemos visto en capítulos anteriores, el impacto de no encontrar un bloque de tamaño 2 o mayor también se propaga a los niveles inferiores dado que la remoción de un subconjunto de asignación de orden  $n$  (con  $n > 1$ ) frecuentemente provoca el posterior desacople de bloques más pequeños. Esto se puede comprobar observando en la tabla 6.2 la fila correspondiente a los bloques de tamaño 1. Si bien los factores de ramificación no acotan la búsqueda de subconjuntos de asignación de orden 1, se aprecia que el método directo encuentra cuatro bloques más que GS-FLCN.

Por último, con respecto a los tiempos de ejecución, el método directo probó ser mucho más eficiente que GS-FLCN por varios órdenes de magnitud. Esto se debe a la naturaleza combinatorial de GS-FLCN, la cual hace que su tiempo crezca en forma exponencial con el tamaño del problema y la profundidad de búsqueda.

Resumiendo, se puede concluir que, para este ejemplo, el método directo demostró ser más eficaz y eficiente que GS-FLCN logrando asignar un 7% más de variables. Al aumentar el tamaño de la submatriz de ocurrencia  $\mathbf{N}$  resultó necesario emplear factores de ramificación para acelerar la ejecución de GS-FLCN, lo cual trajo aparejado una disminución en su capacidad para detectar subconjuntos de asignación.

### 6.3.3. CASO DE ESTUDIO III: PLANTA DE PRODUCCIÓN DE ETANO.

El último ejemplo que analizaremos en este capítulo corresponde a la planta de producción de etano presentada anteriormente (sección 4.4.2.1). Luego de reconfigurar algunas opciones termodinámicas y redefinir parte de la instrumentación, el modelo matemático quedó

formado por un sistema de 1830 ecuaciones con 332 variables medidas y 1600 variables no medidas; constituyendo el caso de estudio de mayor envergadura presentado en esta tesis.

Al igual que en el ejemplo anterior, GS-FLCN fue empleado con factores de ramificación debido a la magnitud del problema. En este caso, por tratarse de un problema mucho más grande que el anterior, resultó necesario acotar aún más la exploración del método para lograr tiempos de cómputo razonables. Se permitió una exploración completa del grafo hasta un nivel de profundidad 5. A partir del nivel 6 se fue acotando gradualmente el espacio de búsqueda, estableciéndose la cota más estricta para todos los niveles mayores que 8.

Los resultados luego de varias iteraciones con ambos métodos se reportan en la tabla 6.3. Observando la última fila de la tabla se aprecia que, mientras el método directo asigna el 66% de las variables, GS-FLCN sólo logra clasificar como observables el 20% de las variables no medidas. La fuerte pérdida de eficacia evidenciada por GS-FLCN en este ejemplo puede atribuirse a dos causas principales. La primera, y la más evidente, es que al tratarse de un problema muy grande fue necesario acotar significativamente la capacidad de exploración del método. Esto obviamente provocó que el método no detectara una gran cantidad de bloques.

Dimensión	1830x1600	
	GS-FLCN	Método directo
1	292	807
2	2	4
3	1	-
4	1	1
5	1	-
6	2	6
7	-	1
11	-	1
16	-	2
22	-	1
27	-	1
35	-	2

39	-	1
pva	20%	66%
tiempo de ejecución (en minutos)	59:23	0:42

**Tabla 6.3.** Resultados del análisis de observabilidad para el caso de estudio III.

La otra razón de este bajo desempeño se debe a las características físicas propias del proceso industrial modelado. Por la planta de producción de etano circulan un total de 12 componentes químicos, esto favorece la conformación de subsistemas grandes dentro del modelo matemático de la planta. En general, cuando mayor es el número de componentes, más grandes serán los subconjuntos de asignación conformados en torno a los balances de masa y componentes de los equipos del proceso. Esto obviamente perjudica a GS-FLCN porque la exploración de los niveles de búsqueda más profundos (los cuales están asociados a estos tipos de subsistemas) suele estar muy acotada a fin de bajar los tiempos de ejecución del método. Si se observa la tabla 6.3, se puede confirmar lo expresado dado que los bloques más grandes detectados por el método directo no pudieron ser encontrados por GS-FLCN. Tal como en el ejemplo anterior, el tiempo de cómputo consumido por GS-FLCN es muy superior al del método directo, lo cual constituye otra fuerte ventaja de este último algoritmo.

Del análisis de este caso de estudio, se extrae como conclusión que, para plantas industriales de gran dimensión, en donde resulta factible la conformación de subconjuntos de asignación grandes, el método directo posee un desempeño fuertemente superior respecto de GS-FLCN. Por tal motivo, en estos casos se recomienda emplear el método directo ó bien particionar la planta en secciones apropiadas, típicamente correspondientes a sectores de planta, y luego realizar el análisis de observabilidad de cada zona usando GS-FLCN. Esta última alternativa puede resultar muy atractiva cuando se desea efectuar un estudio de tipo local, centrado en los equipos más críticos del proceso industrial.

## 6.4. COMENTARIOS FINALES

En este capítulo se presentó un *método directo para análisis de observabilidad*. Está técnica, basada en descomposición de grafos, constituye el último algoritmo de clasificación de variables propuesto en el marco de esta tesis. A diferencia con GS-FLCN y CDHG, el método directo que hemos desarrollado no efectúa búsquedas combinatorias ni emplea reglas heurísticas, efectuando el reordenamiento de la submatriz de ocurrencia  $N$  en dos etapas mediante el uso de bigrafos y digrafos.

La base del algoritmo lo constituye una nueva clasificación de nodos en pareamientos maximales de bigrafos, derivada de la propuesta por Dulmage y Mendelsohn para descomposición de matrices estructuralmente singulares. Para el análisis de subconjuntos prohibidos se diseñó una estrategia basada en tres pasos: test de admisión, reasignación de subconjuntos de asignación y reducción del bigrafo. El test de admisión sigue la política propuesta para GS-FLCN y CDHG. La etapa de reasignación busca evitar la conformación de subconjuntos de asignación prohibidos mediante la permutación entre filas asignadas y redundantes. Por último, la fase de reducción permite la construcción de pareamientos maximales alternativos cuando la reasignación no resulta exitosa o no puede ser aplicada.

El método directo fue comparado con GS-FLCN para tres casos de estudio a fin de evaluar su desempeño en el análisis de observabilidad de plantas industriales. Los resultados alcanzados fueron sumamente satisfactorios logrando asignar en todos los casos una cantidad igual o superior de variables que GS-FLCN. Si bien se comprobó que para problemas pequeños GS-FLCN puede obtener una descomposición de granularidad más fina que el método directo, quedó demostrado que, a medida que el problema crece en dimensión y complejidad, el método

directo logra una mayor eficacia que GS-FLCN. Por otra parte, en lo que refiere a tiempos de cómputo, el método directo mostró ser extremadamente más eficiente que GS-FLCN.

Dejando a un lado el análisis de observabilidad, el método directo constituye una herramienta de particionamiento de matrices muy poderosa y original. La técnica es poderosa, porque puede ser aplicada sobre cualquier tipo de matrices, sin importar su patrón estructural; y original, porque nuestras revisiones bibliográficas no revelaron la existencia de ningún otro algoritmo que permita incorporar restricciones en la conformación de los bloques para guiar el reordenamiento según pautas especificadas. El método es sumamente flexible dado que es posible generar diversos reordenamientos para un mismo problema hasta encontrar el más satisfactorio. Por lo anterior, se puede afirmar que esta nueva técnica constituye una base sólida para el desarrollo de una nueva metodología de resolución de sistemas de ecuaciones no lineales raras, en donde un reordenamiento a FTiB simplificaría la resolución del sistema original, pudiendo ser aplicado incluso sobre sistemas numéricamente singulares.

# ***SISTEMA DE SOPORTE DE DECISIÓN PARA OBSERVABILIDAD***

## **7.1. SISTEMAS DE SOPORTE DE DECISIÓN**

El objetivo global de esta tesis, desde un punto de vista práctico, es el desarrollo de un paquete de software que permita facilitar la labor del profesional en el diseño de instrumentación de plantas. A lo largo de la presente investigación, se estudiaron los principales algoritmos de observabilidad empleados en la actualidad, proponiéndose nuevos métodos con el fin de obtener mejores desempeños. En los capítulos anteriores se expusieron en detalle las características fundamentales de las metodologías propuestas, sus alcances y limitaciones.

Sin embargo, una herramienta computacional capaz de asistir al ingeniero en esta tarea debe considerar otros aspectos adicionales, aparte del algoritmo de observabilidad en sí, que contribuyan a facilitar el diseño de instrumentación. En tal sentido, la amigabilidad del software, los chequeos de consistencia y seguridad de los datos ingresados, la generación automática de reportes y la capacidad de organizar, analizar y visualizar la información de forma útil resulta fundamental para el usuario, quien puede así tomar mejores decisiones de diseño al poder apreciar conjuntamente todos los factores que intervienen en la instrumentación de una planta compleja.

Dadas las características de esta aplicación, se llegó a la conclusión de que un **sistema de soporte de decisión** (*decision support system, DSS*) resultaba el tipo de software más apropiado para abordar esta problemática. Básicamente, los DSSs son sistemas basados en tecnologías computacionales coherentes (hardware, software y documentación de soporte) que asisten al usuario en la toma de decisiones semiestructuradas. Los DSS se focalizan en mejorar la efectividad en la toma de decisiones complejas, en donde la diversidad de aspectos a tener en cuenta y la forma en que estos se interrelacionan tornan muy difícil la labor del especialista, quien habitualmente debe elegir entre distintas opciones igualmente factibles. En general, las tareas abordadas por este tipo de software involucran el manejo de una gran cantidad de combinaciones de datos y posibilidades o caminos de decisión alternativos, los cuales hacen imposible un tratamiento completo y confiable por parte de la mente humana sin que se escapen situaciones especiales o factores semiocultos. Básicamente, el DSS consiste de un sistema de manejo de la información al cual se le agregan herramientas de software que ayudan en la toma de decisiones. Las primeras aplicaciones surgieron dentro del campo de la economía, colaborando en las decisiones a nivel gerencial.

Para lograr DSS efectivos, los diseñadores de estos programas interactúan con los potenciales usuarios del software con el fin de determinar con precisión los mecanismos y procedimientos que intervienen en el proceso de toma de decisiones llevados a cabo por estas personas. La meta es interpretar y volcar ese conocimiento en el desarrollo de herramientas computacionales que procesen y organicen la información de modo útil para ayudar en la toma de decisiones. Los principios y pautas generales para el diseño y desarrollo de DSSs se presentan en Bennett (1983).

### **7.1.1. DSSs EN INGENIERÍA**

En el campo de la ingeniería de procesos, las tendencias actuales conducen al desarrollo de paquetes computacionales de tipo "plant-wide" – es decir, capaces de analizar una planta de procesos químicos en su totalidad en forma conjunta. Un soporte computarizado eficiente es indispensable para aplicaciones industriales de gran escala dado que el problema crece enormemente incrementándose la complejidad en forma significativa. Durante la pasada década, se generaron herramientas "inteligentes" para diversos propósitos. Hartmann y *co.* (1990) presentaron un DSS interactivo basado en modelos para el análisis y síntesis de procesos químicos, con un módulo para optimización basada en multicriterios. Por su parte, Lygeros y *co.* (1996) desarrollaron un DSS para integración de sitios y diseño de utilidades. En Argentina, Leone (1996) diseñó un sistema basado en el conocimiento para ayudar en la identificación y evaluación de seguridad de riesgos en plantas de procesos y en el reconocimiento de problemas de operabilidad. En Ingeniería Química, también han sido desarrolladas aplicaciones dentro de la tecnología de sistemas expertos. Por ejemplo, Bugaeva y *co.* (1996) reportó un sistema experto basado en reglas para diseñar y simular procesos de purificación de gas, mientras que Tsai y Chang (1995) emplearon redes neuronales para detección de fallas y diagnósticos en procesos dinámicos.

En particular, el campo del análisis de observabilidad puede beneficiarse enormemente con el desarrollo de un DSS dado que la elección de los lugares para la ubicación de los instrumentos en una planta de procesos requiere del análisis sistemático de grandes cantidades de información heterogénea. Un sistema experto no resulta apropiado para esta aplicación específica. La razón principal es que obtener en forma completamente automática los resultados finales, sin que el usuario intervenga en la toma de las decisiones claves durante el diseño de la instrumentación, conduciría a soluciones poco confiables debido a que muchas decisiones involucradas en el análisis son subjetivas y muy dependientes de cada caso de estudio en particular. En contraste, un DSS resulta más conveniente dado que permite al ingeniero de procesos, quien interactúa con el software, jugar un rol más activo durante el diseño de instrumentación. Para esto, el DSS debe contar con herramientas que analicen y organicen la información de forma tal que el usuario pueda tomar las decisiones centrales del diseño teniendo una visión global e integrada de todos los aspectos que intervienen en el problema.

## **7.2. DISEÑO DEL DSS**

En este capítulo se presenta el desarrollo de un DSS confiable y amigable para el análisis de observabilidad de plantas de procesos (Ponzoni *y co.*, 2000), considerándose varios objetivos de diseño. El software debía permitir el modelado riguroso y preciso de procesos reales. El paquete también debía resultar eficiente y robusto para el tratamiento de problemas de gran dimensión, conduciendo a soluciones de buena calidad. Por último, el DSS debía ser flexible con respecto a la adición de algoritmos de observabilidad alternativos y/o herramientas para asistir en la toma de decisiones.

### **7.2.1. ESTRUCTURA GENERAL DEL DSS**

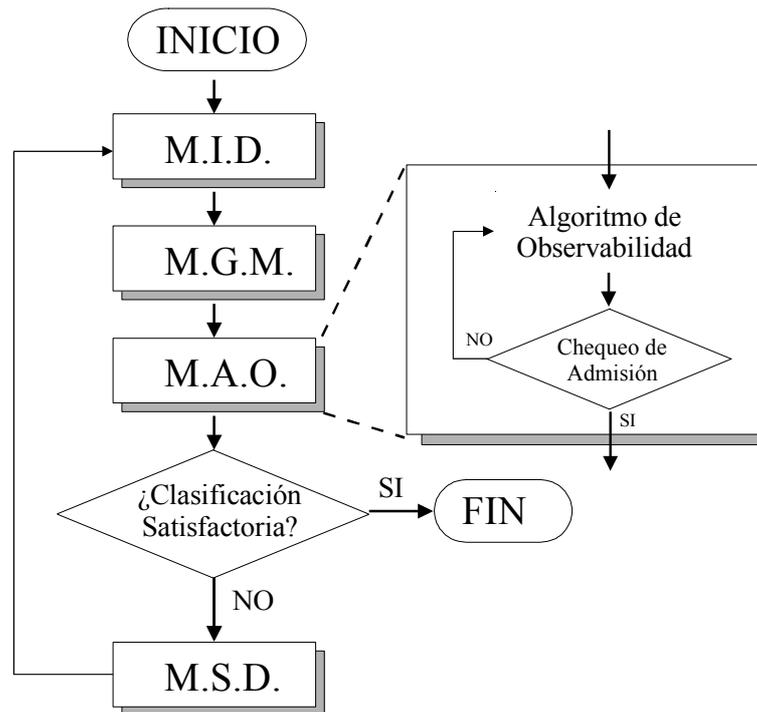
El DSS implementado consiste de una interface inteligente, un generador de modelos flexible (**MGM**), un módulo robusto para análisis de observabilidad (**MAO**) que clasifica las variables no medidas y un módulo para soporte de decisiones (**MSD**) que provee herramientas para el análisis de los resultados. La interface comprende dos partes principales: especificación del problema (**MID**) y reportes de la solución (información de salida) para cada modulo. El paquete permite visualizar los resultados obtenidos por el algoritmo de observabilidad y brinda diversos reportes para ayudar en la toma de decisiones.

En la figura 7.1 se presenta un esquema de los principales componentes del software y como estos interactúan con el DSS. El software fue implementado para correr bajo plataformas Windows 98/NT. La interface y el **MGM** fueron implementados en Visual Basic, mientras que los módulos **MAO** y **MSD** fueron codificados en Visual C++. La interface y el **MGM** fue desarrollado sobre la base de ModGen, un software para generación de modelos matemáticos de plantas, que había sido codificado en Visual Basic. El resto del software se decidió implementarlo en C por razones de compatibilidad con plataformas UNIX.

### **7.2.2. INTERFACE: MÓDULO DE INGRESO DE DATOS (MID)**

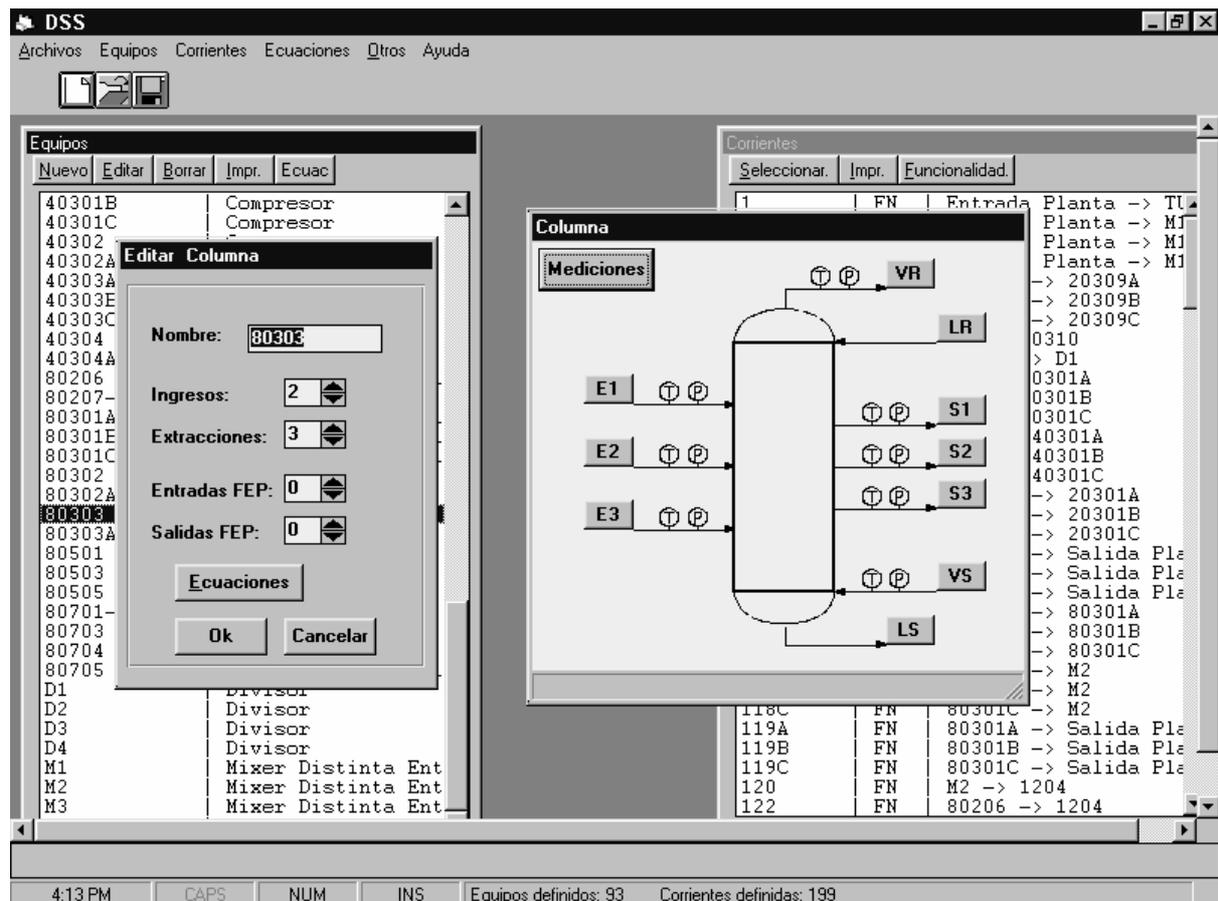
La información requerida por el procedimiento de clasificación proviene del modelo matemático que representa el funcionamiento de la planta. El modelo es generado sobre la base de la topología del proceso y un conjunto inicial de mediciones. Como ya hemos explicado al comienzo de esta tesis, la topología de una planta industrial puede ser visualizada como un grafo dirigido cuyos nodos y aristas representan unidades de procesos (tales como reactores, bombas y válvulas) y corrientes respectivamente. Las direcciones de las aristas indican el sentido de circulación de los fluidos a través de la planta. En esta interface, el usuario especifica

interactivamente los equipos junto con sus corrientes y mediciones asociadas, usando el formato de presentación que típicamente emplean los ingenieros de procesos para sus esquemas de planta. Por ejemplo, la figura 7.2 muestra la definición de una columna. Las interfaces correspondientes a las restantes categorías de equipos se muestran en el apéndice que se presenta al final de la tesis.



**Figura 7.1.** Principales componentes del DSS.

La información de entrada también incluye alternativas de modelado para la selección de diferentes niveles de complejidad con respecto no sólo a la cantidad de ecuaciones involucradas, sino también a las funcionalidades consideradas para cada propiedad termodinámica. También resulta posible introducir ecuaciones adicionales en términos simbólicos. El paquete automáticamente efectúa chequeos para consistencia de datos, rechazando especificaciones contradictorias o configuraciones físicamente ilógicas.



*Figura 7.2. Definición de un equipo tipo columna.*

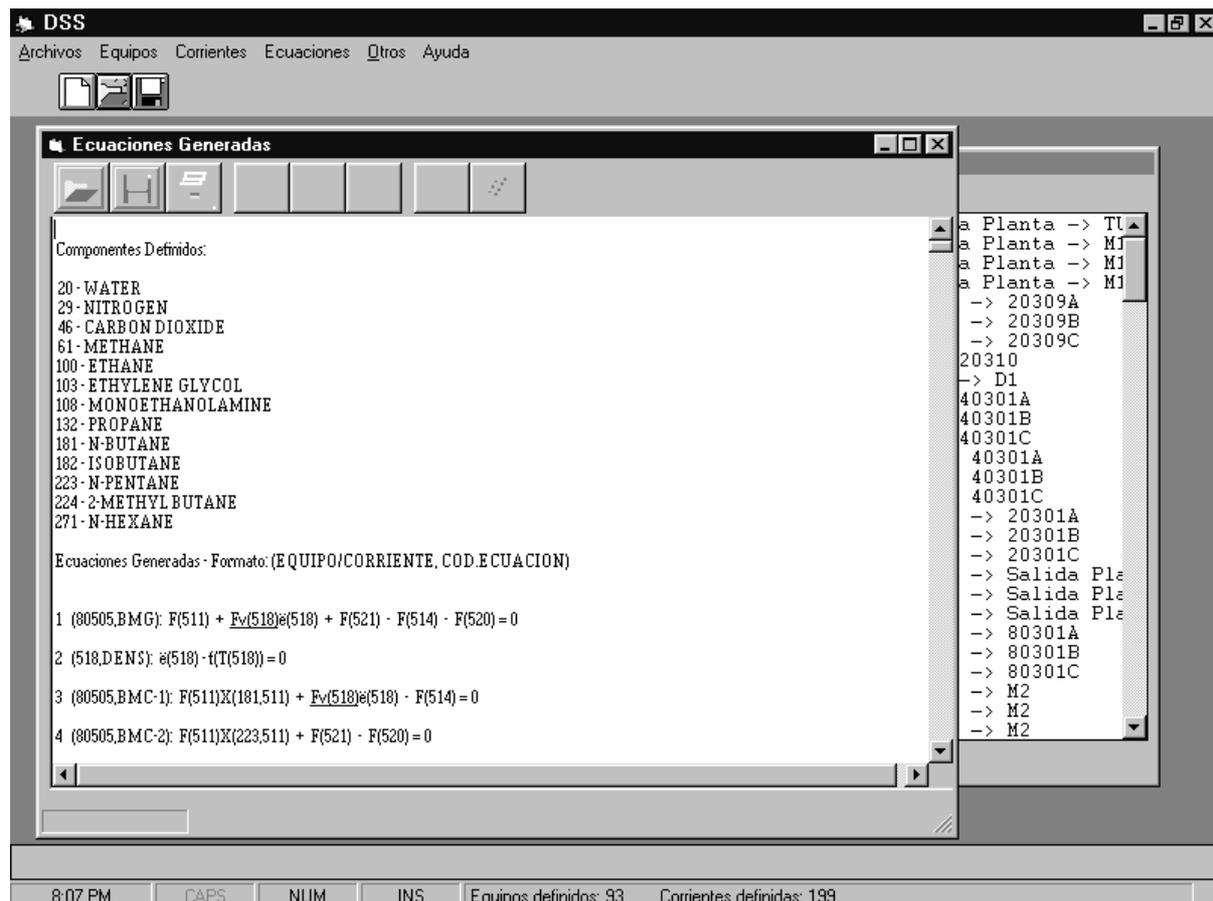
El software fue concebido para permitir amplias posibilidades de modificación de los datos de entrada. Las unidades de proceso pueden ser agregadas o removidas fácilmente a fin de redefinir la topología de la planta. Asimismo, la localización de los instrumentos dentro de la planta puede ser cambiada de forma muy simple durante cualquier etapa del diseño. Estas características son notablemente importantes por dos razones principales. Primero, la descripción de la topología de una planta debería ser un procedimiento muy ágil dado que los problemas industriales generalmente involucran una gran cantidad de equipos y corrientes. Las modificaciones en la topología se necesitan con frecuencia en este contexto, tanto para analizar eficientemente configuraciones alternativas para una misma planta como para determinar la mejor representación de un proceso. Más aún, con respecto al proceso de análisis de

observabilidad en sí, cada iteración principal intermedia siempre requiere la adición o remoción de mediciones.

### **7.2.3. MÓDULO DE GENERACIÓN DE MODELOS (MGM)**

Este módulo construye un modelo matemático del proceso industrial en estudio usando la información de entrada provista por el usuario a través del **MID**. Una misma planta puede ser representada en varias maneras dado que es posible generar modelos con diferentes niveles de complejidad. Cada modelo provee una particular aproximación al comportamiento real del proceso. En términos generales, cuando más complejo es el modelo empleado, más precisa se torna la representación. De este modo, el usuario debe decidir cuál es la caracterización más conveniente para el proceso, con el objeto de tener el suficiente conocimiento de la planta con una complejidad razonable.

En lo que refiere a la estructura del modelo matemático, el módulo **MGM** construye conjuntos de ecuaciones algebraicas fundamentales que representan balances de masa y energía en estado estacionario y relaciones termodinámicas. Al final de la tesis se presenta un apéndice en donde se describen en detalle las diferentes clases de ecuaciones generadas por el **MGM** para cada tipo de equipo. La figura 7.3 muestra una pantalla en donde se aprecia una pequeña parte del modelo matemático generado para la planta de producción de etano analizada como caso de estudio a largo de esta tesis. Una vez que el modelo matemático ha sido definido, el módulo genera la matriz de ocurrencia **N** (ver figura 3.2) correspondiente a las variables no medidas y otros archivos auxiliares requeridos por el **MAO**.



*Figura 7.3. Modelo matemático generado automáticamente por el MGM.*

#### 7.2.4. MÓDULO DE ANÁLISIS DE OBSERVABILIDAD (MAO)

Las herramientas de clasificación elegidas para este DSS fueron GS-FLCN y el método directo, los cuales fueron descritos en los capítulos 4 y 6 de esta tesis. Se eligieron estas técnicas porque probaron ser las metodologías más apropiadas para el análisis de plantas industriales complejas de gran dimensión. No obstante, el DSS fue concebido para permitir una fácil incorporación de módulos de observabilidad alternativos a fin de obtener la clasificación mediante otras técnicas estructurales y así lograr un software más flexible.

Como se describió anteriormente, el procedimiento completo de clasificación es iterativo y usualmente requiere varias ejecuciones del algoritmo de observabilidad. Después de cada corrida, el ingeniero inspecciona los subconjuntos de asignación, llevando adelante un chequeo

de admisión sobre la base de la interpretación física de las ecuaciones involucradas. Este chequeo ayuda a detectar subsistemas numéricamente singulares, los cuales deben ser rechazados debido a que no tienen solución única. En esta etapa, los bloques no deseables son incorporados a la lista de subconjuntos prohibidos a fin de que estos sean descartados inmediatamente cada vez que aparezcan en subsecuentes iteraciones. Si GS-FLCN localiza uno de estos subconjuntos durante las siguientes iteraciones, el bloque es automáticamente rechazado.

Una clasificación de variables es satisfactoria para un dado conjunto de mediciones cuando GS-FLCN obtiene un reordenamiento donde todos los subconjuntos de asignación son aceptables desde un punto de vista ingenieril. Una vez que esto ha sido logrado, es posible pasar a la siguiente etapa del análisis en donde el usuario toma decisiones respecto de la remoción o agregado de sensores a la planta.

#### **7.2.5. MÓDULO DE SOPORTE DE DECISIÓN (MSD)**

Este módulo contiene las herramientas de análisis que ayudan al usuario en la toma de decisiones inteligentes a fin de obtener una configuración final que posea un número reducido de instrumentos ubicados en los lugares más convenientes, proveyendo la cantidad de información necesaria y suficiente para conocer el estado real de la planta en forma completa y confiable. En este punto, el ingeniero ya ha obtenido una clasificación aceptable del **MAO**. De la figura 3.2 resulta claro que las variables clasificadas como no observables serán indeterminables, es decir, no podremos conocer sus valores. Por tal motivo, es fundamental que el usuario determine cuáles variables no medidas son de interés. Sólo se alcanzará una clasificación definitiva cuando todas las variables de interés sean observables.

Es importante remarcar que, dada una clasificación donde algunas variables de interés son indeterminables, no resulta aconsejable medir todas ellas para volverlas observables ya que esto derivaría en una configuración con una cantidad excesiva de mediciones. Esto contradice el objetivo de lograr un buen conocimiento del estado de la planta con un número reducido de mediciones. Por otro lado, tampoco se recomienda la incorporación o remoción masiva de sensores. La razón por la cual esta práctica resulta desaconsejable se justifica teniendo en cuenta que cada cambio en el conjunto de mediciones genera una partición  $[N|M]$  diferente de la matriz de ocurrencia  $O$ . Luego, el **MAO** debe ser ejecutado después de unas pocas modificaciones en las mediciones. En este sentido, las herramientas de análisis ayudan al usuario a predecir el nivel de desacople que provocará el agregado de una medición en particular, es decir la cantidad de variables que se tornarán observables por el agregado de una dada medición. En otras palabras, el usuario puede establecer cuánto conocimiento adicional obtendrá a partir de la incorporación de un nuevo instrumento. Sin embargo, la decisión final suele ser compleja e implica asimismo considerar relaciones costo-beneficio que requieren del criterio experto del ingeniero. En la práctica, cada decisión estará basada en los resultados del análisis explicado en esta sección, junto con otro conjunto de criterios no estructurados que responden a políticas de gerenciamiento de plantas y estrategias de instrumentación propias de cada empresa.

Una de las principales herramientas de análisis provistas por el DSS consiste en la determinación automática del tipo y nivel de desacople que puede inducir la medición de una variable no observable. En otras palabras, para cada variable no observable  $v_i$ , es posible determinar un grupo  $d_i$  de variables no observables que se volverían observables si  $v_i$  fuera medida. Así,  $d_i$  constituye el conjunto de desacople de la variable  $v_i$ . Los conjuntos de desacople pueden ser determinados fácilmente siguiendo la secuencia natural de resolución de ecuaciones,

determinando qué variables originalmente no observables pueden ser calculadas en términos del nuevo conjunto de mediciones obtenido como consecuencia de mover la columna  $v_i$  desde  $\mathbf{N}$  a  $\mathbf{M}$ .

Por ejemplo, si la ecuación  $e$  en la submatriz de tamaño  $n \times n$  sólo contiene dos variables no observables:  $v_i$  y  $v_j$ , entonces  $v_j$  formará parte de  $d_i$ . Esto es consecuencia de que, como  $v_i$  ha sido medida, la variable  $v_j$  ahora puede ser calculada de la ecuación  $e$ . En este caso, diremos que  $v_i$  y  $v_j$  son *variables directamente dependientes* una de otra. Luego, el conjunto de desacople puede ser construido transitivamente. Dado que  $v_j$  se torna observable, todas las variables  $v_k$  (para  $k=1, \dots, p$ ;  $k \neq j$ ;  $k \neq i$ ) que dependen directamente de  $v_j$  serán observables como consecuencia indirecta de la medición de  $v_i$ . Luego, las  $v_k$  variables también serán incorporadas en  $d_i$ .

La información vinculada a los conjuntos de desacople calculados por el DSS puede ser empleada en diferentes formas para inferir qué sensores deben ser agregados o removidos. Un criterio básico puede ser agregar el instrumento que provea el mayor nivel de desacople dado que esto tiende a reducir el número de sensores. No obstante, otros conjuntos de desacople pueden ser más promisorios debido a que contienen una gran cantidad de variables de singular importancia. Dentro de este razonamiento, el ingeniero puede definir un conjunto de variables indeterminables de interés  $\mathbf{I}$ . Para problemas de gran escala, los elementos en  $\mathbf{I}$  deberán corresponder preferentemente a diferentes sectores de planta. Si  $\mathbf{I}$  ha sido especificado, se puede automatizar fácilmente la búsqueda de un subconjunto reducido de variables que deberían ser medidas a fin de volver conocidas todas las variables en  $\mathbf{I}$ . Esta tarea puede ser implementada en forma sencilla mediante operaciones elementales de intersección de conjuntos. Cada conjunto de desacople es intersectado con  $\mathbf{I}$ , eligiéndose como nueva medición a aquella variable tal que la intersección entre su conjunto de desacople e  $\mathbf{I}$  tenga mayor cantidad de

elementos. Luego, todas las variables en común son removidas de  $\mathbf{I}$  dado que éstas serán observables a partir del agregado del nuevo sensor. Este procedimiento es repetido hasta que  $\mathbf{I}$  quede vacío. En este punto, se ha elegido un conjunto reducido de variables cuya medición resulta suficiente para conocer todas las variables indeterminables de interés. En otras palabras, el programa selecciona un número mínimo de conjuntos de desacople tales que la unión de los mismos incluye todas las variables presentes en  $\mathbf{I}$ .

### **7.3. UN CASO DE ESTUDIO INDUSTRIAL**

Para ilustrar el desempeño de este DSS como herramienta para agilizar el manejo de los resultados de la clasificación, se presenta en esta sección una fase completa de diseño para la planta de producción de etano presentada en el capítulo 4 de esta tesis, cuyo modelo matemático riguroso está integrado por un conjunto de 1830 ecuaciones algebraicas fuertemente no lineales.

#### **7.3.1. CLASIFICACIÓN INICIAL**

La primera etapa del diseño de instrumentación consiste en cargar los equipos y corrientes que conforman la topología del proceso, configurar el conjunto inicial de mediciones y seleccionar las opciones generales de modelado. Para este caso de estudio, se incorporó un grupo inicial muy reducido de mediciones con el fin de evitar sensores redundantes que incrementarían innecesariamente los costos de instrumentación. Al finalizar esta primera etapa, el problema contenía 1600 variables no medidas. Luego, se generó automáticamente el modelo matemático correspondiente, ejecutándose después el algoritmo de observabilidad GS-FLCN. Durante el chequeo de admisión se prohibieron más de 50 subconjuntos de asignación. Los bloques prohibidos más pequeños correspondían en su mayoría a subsistemas lineales

cuadrados deficientes en rango correspondientes a variables del proceso que siempre tenían el mismo valor para corrientes distintas. Los subconjuntos rechazados de mayor tamaño generalmente estaban conformados por los balances de masa para todas las componentes de la mezcla y el balance de masa global para un determinado equipo o corriente. Esto producía que una ecuación fuese combinación lineal de las restantes, lo cual derivaba en subsistemas sin solución única.

Una vez obtenida una clasificación libre de subconjuntos prohibidos, el patrón resultante quedó conformado por 333 variables observables y 1267 variables no observables. Esta clasificación necesitaba posteriores refinamientos debido a que muchas variables de interés permanecían como indeterminables.

### ***7.3.2. ETAPA DE TOMA DE DECISIONES***

En esta fase del diseño, el ingeniero determina qué instrumentos conviene agregar a fin de obtener toda la información requerida del proceso. Uno de los equipos críticos de la planta de producción de etano es la demetanizadora situada en el área criogénica. De hecho, existen dos sectores mellizos con una columna demetanizadora en cada uno (ver figura 4.4). Dada la importancia de estos equipos se adicionaron mediciones para tener un conocimiento completo de ambos. Además, el usuario agregó mediciones de temperatura y caudales de masa para algunas corrientes de entrada a la planta elegidas adecuadamente de forma tal que proveyeran los mayores niveles de desacople de variables utilizando los valores de los conjuntos de desacople brindados por el DSS. También se introdujeron dos sensores en otros sectores de la planta dado que estas mediciones proveían comparativamente un alto grado de desacople. En total se incorporaron 12 nuevos instrumentos como resultado de la primera iteración de diseño completa efectuada con el **MSD**.

Los nuevos sensores fueron agregados a través del **MID** y posteriormente se actualizó el modelo utilizando el **MGM**. Luego, se ejecutó el algoritmo de observabilidad del **MAO** sobre la nueva matriz de ocurrencia hasta obtener una nueva clasificación sin conjuntos prohibidos. Como consecuencia de la adición de los nuevos sensores, la nueva descomposición contenía 22 variables observables más y el número de indeterminables se redujo en 34. Como todavía restaba determinar un número importante de variables de interés, el **MSD** fue empleado nuevamente siguiendo las mismas pautas generales de diseño explicadas anteriormente. Luego de siete iteraciones se alcanzó una clasificación final satisfactoria.

#### **7.4. COMENTARIOS FINALES**

En términos generales, en este capítulo se ha presentado un Sistema de Soporte de Decisión especialmente desarrollado para aplicaciones de diseño de instrumentación. El DSS fue concebido como una herramienta computacional de ayuda a los ingenieros en la etapa crucial del análisis de observabilidad, la cual constituye la parte que consume más tiempo en el proceso de diseño. El software fue diseñado a medida con el fin de satisfacer los requerimientos del usuario en cuanto a interfaces amigables, consistencia de datos, flexibilidad del modelo matemático, fácil visualización de los resultados clave del análisis, confiabilidad y eficiencia en tiempo de ejecución.

Un aspecto central del trabajo fue el diseño de herramientas para ayudar en la toma de decisiones, los cuales fueron creadas para proveer alternativas altamente dedicadas para el análisis de resultados. De este modo, se logró plasmar en un paquete de software muchos de los conocimientos y desarrollos llevados adelante durante esta tesis, lográndose un producto final muy versátil y confiable que permitirá agilizar el procedimiento de clasificación con significativos ahorros de tiempo.

Finalmente, cabe destacar que el DSS propuesto para análisis de observabilidad puede ser empleado en el futuro como base para obtener un paquete de software para el diseño completo de instrumentación de plantas. Esto no es trivial, pues resulta necesario generar conocimiento en el tema de análisis de redundancias, a fin de diseñar un algoritmo estructural compatible con nuestro enfoque en robustez y calidad de resultados, de modo que pueda ser incorporado al DSS con el fin de lograr una clasificación de variables completa. Esta nueva técnica, junto con una metodología para reconciliación de datos de planta que permita ajustar los valores de las mediciones, podrían ser incorporadas a nuestro DSS con lo cual se obtendría una herramienta computacional poderosa y completa, de gran utilidad para la actual industria de procesos.

## *CONCLUSIONES E INVESTIGACIONES FUTURAS*

El objetivo de esta tesis ha sido desarrollar nuevas técnicas en el campo del análisis de observabilidad en procesos industriales aplicando teoría de grafos y conceptos avanzados de ciencias de la computación, los cuales van desde el uso de metodologías para el diseño y desarrollo de paquetes de software hasta la utilización del paralelismo como alternativa para mejorar el desempeño de los algoritmos propuestos.

Como resultado de estas investigaciones, se ha logrado el desarrollo de técnicas robustas y eficientes especialmente diseñadas para la clasificación de variables no medidas en procesos industriales con modelos matemáticos complejos, los cuales requieren la definición de ecuaciones altamente no lineales. De este modo, problemas que antes no podían ser tratados con los métodos de observabilidad clásicos, o que requerían una fuerte simplificación de su modelo

matemático para ser resueltos, pueden ahora ser tratados en forma rigurosa, precisa y confiable mediante el empleo de los nuevos algoritmos propuestos en esta tesis.

Con el objeto de brindar al ingeniero de planta un software más especializado, se diseñó e implementó un sistema de soporte de decisión que engloba toda la experiencia adquirida en clasificación de variables. Fue posible generar una herramienta computacional, eficiente y robusta que es capaz de asistir al usuario en forma versátil y confiable en la compleja tarea de establecer la ubicación más apropiada de los sensores necesarios para el control del correcto funcionamiento de una planta real. El paquete posibilita analizar en forma rigurosa plantas de cualquier dimensión, incluso las de gran envergadura.

En este capítulo se resumen los principales resultados obtenidos con cada una de las técnicas desarrolladas, el alcance de los nuevos algoritmos y se sintetizan las comparaciones entre las distintas propuestas. Además, se presentan recomendaciones y lineamientos generales para futuras investigaciones en el tema, así como también sugerencias sobre extensiones naturales del trabajo desarrollado en esta tesis.

## **8.1. CONCLUSIONES**

El análisis de observabilidad de las variables involucradas en los modelos matemáticos correspondientes a plantas industriales constituye una herramienta esencial para mejorar el diseño de instrumentación y por ende, el control de procesos industriales. Dada la complejidad involucrada en estas tareas, resulta imprescindible contar con paquetes de software capaces de asistir al profesional en la toma de decisiones propia de este tipo de problemas. Un programa de tales características debe estar constituido por dos partes fundamentales. En primer lugar, un algoritmo eficiente y robusto que provea una clasificación de variables correcta. Por otra parte,

también es necesaria una serie de herramientas auxiliares que brinden en forma dinámica y versátil información útil y confiable para tomar decisiones que conlleven a una buena elección de los mejores lugares en donde agregar o remover instrumentación en el proceso en estudio.

Con respecto al análisis de observabilidad, el estudio crítico de las metodologías existentes para sistemas de ecuaciones no lineales reveló grandes deficiencias tanto en robustez como en rango de aplicación, las cuales impiden su empleo en plantas industriales con modelos matemáticos complejos. Por esta razón, se diseñaron e implementaron distintas técnicas con el objeto de superar las limitaciones mencionadas.

El primer algoritmo propuesto, al que denominamos GS-FLCN, surgió sobre la base de la metodología de la columna dominante (DCB), la cual era la técnica existente más apropiada para modelos no lineales. DCB es una técnica combinatorial muy eficiente en cuanto a tiempos de ejecución. Su principal desventaja es la falta de robustez en el tratamiento de casos industriales grandes con presencia de reciclos. Esto se debe a que algunos subsistemas observables del modelo matemático no pueden ser detectados, lo cual puede conducir a clasificaciones incompletas o erróneas.

A diferencia de su antecesor, GS-FLCN logra un análisis de observabilidad robusto. La nueva técnica emplea un grafo dirigido  $G$  para representar las relaciones existentes entre las distintas variables y ecuaciones que modelan el proceso. Luego, todos los subsistemas de variables observables son detectados mediante la exploración de  $G$  siguiendo una heurística especial para el recorrido que da prioridad a los nodos menos conectados del grafo. La técnica es esencialmente recursiva y genera un espacio de búsqueda arbóreo.

GS-FLCN fue empleado en el análisis de varios casos industriales y académicos, lográndose clasificaciones muy superiores a las alcanzadas usando DCB. No obstante los buenos resultados obtenidos, se observó que los tiempos de ejecución eran altos para problemas

de tamaño industrial con presencia de subconjuntos de asignación de gran dimensión. Esto motivó el estudio y la propuesta de distintas alternativas que mejoraran el desempeño del algoritmo.

De este modo, surgió en primera instancia la idea de "podar" algunas ramas del espacio de búsqueda con el fin de acelerar la exploración del grafo. La técnica así derivada se denominó GS-FLCN con factores de ramificación (BF). Básicamente, los BF acotan el número de aristas a visitar desde cada nodo, reduciendo la amplitud del espacio de búsqueda. Esta variante posee muy buen desempeño ya que posibilita ajustar el grado de exploración para cada tamaño de subconjunto de asignación permitiendo de este modo concentrar la búsqueda en las regiones de mayor interés.

Posteriormente se desarrolló una versión paralela-distribuida del algoritmo, la cual también incluye los factores de ramificación. Este nuevo algoritmo, al que denominamos GS-pFLCN, posee una estrategia de paralelización que sigue un modelo MASTER-WORKER, aplicándose el paradigma de descomposición de dominios sobre el espacio de búsqueda arbóreo. La implementación del método se efectuó utilizando la librería de pasaje de mensajes PVM (Parallel Virtual Machine).

GS-pFLCN fue utilizado para el análisis de dos casos de estudio industriales de mediano tamaño, lográndose excelentes ganancias en lo que refiere a tiempos de ejecución con respecto a su versión secuencial. En vista de los resultados obtenidos, se recomienda fuertemente la aplicación de GS-pFLCN para el análisis de procesos industriales de gran envergadura, dado que la infraestructura necesaria para ejecutar este programa sólo requiere de una red de área local de estaciones de trabajo, presentes en cualquier industria actual, y el ahorro en tiempo de corrida es muy significativo.

Otro enfoque alternativo fue el diseño de métodos no combinatoriales. El primero de ellos, denominado algoritmo de detección de ciclos en hipergrafos (CDHG, por sus siglas en inglés), utiliza un hipergrafo como medio de representación de la conexión existente entre las variables y ecuaciones del modelo. La implementación efectuada para este algoritmo utiliza seis reglas heurísticas e incorpora la variante simple de simulated annealing para efectuar distintos recorridos del hipergrafo a fin de detectar todos sus ciclos. Cada uno de estos ciclos tiene asociado un subconjunto de asignación, es decir, un subconjunto de variables observables. CDHG fue empleado en varios casos de estudio, obteniéndose buenas clasificaciones con bajos tiempos de ejecución. Aunque la robustez del método no es mala, se aprecia que la calidad de los resultados alcanzados es fuertemente dependiente del conjunto de heurísticas utilizado. En otras palabras, distintos tipos de procesos industriales pueden requerir de diferentes grupos de heurísticas, dado que una heurística exitosa para una planta en particular puede perder efectividad cuando se la aplica a otro problema. Por lo tanto, se decidió integrar CDHG con GS-FLCN en una metodología que denominamos Método Híbrido, la cual buscaba combinar las virtudes de ambos algoritmos. La filosofía seguida fue emplear CDHG como un preprocesamiento para la aplicación posterior de GS-FLCN. De este modo, el problema original es reducido rápida y significativamente por CDHG, para luego aplicar GS-FLCN sobre la parte del problema correspondiente a las variables que CDHG no pudo clasificar. Al aplicar GS-FLCN sobre de un matriz reducida se logran mejores tiempos de ejecución, eliminándose o reduciéndose el nivel de podas requerido en casos muy grandes.

La técnica combinada se empleó en el análisis de observabilidad de dos problemas industriales grandes correspondientes al revamp de una planta de producción de etano y al diseño de instrumentación de una planta de síntesis de amoníaco. En ambos casos los resultados obtenidos fueron superiores a la aplicación de CDHG en forma individual. Esto permite extraer

la siguiente conclusión: aunque la aplicación aislada de CDHG puede resultar insuficiente para el análisis de observabilidad de plantas grandes, el mismo puede ser empleado provechosamente como técnica de preprocesamiento para algoritmos de clasificación de variables más robustos en virtud de sus bajos tiempos de ejecución.

La última de las técnicas diseñadas en esta tesis es un método directo basado en descomposición de bigrafos y digrafos. El corazón de este algoritmo es una nueva clasificación de nodos en pareamientos maximales de bigrafos, derivada de la metodología propuesta por Dulmage y Mendelsohn para la descomposición de matrices estructuralmente singulares.

El manejo de restricciones fue incorporado mediante una estrategia de tres pasos: test de admisión, reasignación de subconjuntos de asignación y reducción del bigrafo. El test de admisión es equivalente a los empleados en GS-FLCN y CDHG. La fase de reasignación evita la construcción de subconjuntos de asignación prohibidos permutando filas asignadas con filas redundantes. Por último, la etapa de reducción genera nuevos pareamientos maximales para los casos en donde la reasignación no tuvo éxito o no puede ser empleada.

Tanto el método directo como GS-FLCN fueron empleados para efectuar la clasificación de variables de varias plantas industriales a fin de comparar sus desempeños. Los resultados obtenidos por el primero fueron excelentes, pudiéndose asignar en todos los casos una cantidad de variables igual o superior a la lograda por GS-FLCN. En general se observó que, si bien para problemas pequeños GS-FLCN puede obtener una descomposición de granularidad más fina que el método directo, a medida que el problema crece en dimensión y complejidad este último logra una mayor eficacia que GS-FLCN. Asimismo, en cuanto a tiempos de cómputo, el método directo mostró ser significativamente más eficiente que GS-FLCN.

En resumen, dadas las características y resultados obtenidos por ambos algoritmos, se recomienda la utilización de GS-FLCN para el análisis de observabilidad de plantas de pequeño

tamaño o sectores de planta reducidos, porque obtiene una mejor descomposición del sistema de ecuaciones. En cambio, si se trabaja con plantas de tamaño mediano o grande, se aconseja usar el método directo, dado que este resulta más eficaz y eficiente para estos casos.

Finalmente, se desarrolló un sistema de soporte de decisión (DSS) para análisis de observabilidad en plantas de procesos tomándose como algoritmos de clasificación el método directo y GS-FLCN con factores de ramificación. En esta etapa, resultó fundamental el diseño de herramientas específicamente ideadas para ayudar al usuario en la toma de decisiones.

Dado el gran volumen de información heterogénea involucrada en el análisis, sumado al alto grado de interrelación existente entre los distintos datos y la complejidad inherente de los modelos matemáticos empleados para representar con fidelidad el funcionamiento de los procesos industriales, resultó imprescindible desarrollar programas de soporte que extrajeran, cruzaran y procesaran los datos a fin de que el usuario contara con la información realmente relevante para la toma de decisiones, sin distraerse con aspectos secundarios o interpretando datos de escaso valor para el problema al resolver.

La principal herramienta propuesta emplea los denominados conjuntos de desacople (CD). Esta técnica permite predecir el impacto de remover o agregar una nueva medición a la configuración actual de la planta bajo estudio. Los CD sirven tanto para el análisis individual de cada variable indeterminable, como para la detección automática del mínimo conjunto de mediciones a agregar. De modo que un conjunto preseleccionado de variables indeterminables de interés para el usuario se torne observable en la siguiente iteración.

En el desarrollo del paquete de software también se consideraron otros aspectos tendientes a automatizar todas aquellas tareas engorrosas vinculadas al análisis, tales como chequeos de consistencia en la especificación del modelo, visualización amigable de los subsistemas correspondientes a los subconjuntos de asignación. En tal sentido, un objetivo de

diseño fue facilitar los chequeos de admisión y la generación de reportes parciales y finales durante el proceso de análisis de observabilidad.

A modo de conclusión final, se puede afirmar que el DSS desarrollado constituye una valiosa ayuda para el ingeniero de procesos en la compleja tarea que implica el análisis de observabilidad de plantas industriales. El DSS sintetiza así buena parte del conocimiento generado en este trabajo de tesis, brindando un producto tecnológico, tangible y concreto, como resultado de estos años de investigación.

## **8.2. INVESTIGACIONES FUTURAS**

A futuro se vislumbran interesantes perspectivas para continuar las investigaciones y resultados alcanzados en este trabajo de tesis. Por un lado, podría abordarse el estudio y desarrollo de metodologías que permitan obtener una configuración apropiada de la localización inicial de las mediciones. Actualmente, cuando el ingeniero ingresa la topología de la planta al DSS, también indica la configuración inicial de la instrumentación a partir de la cual comenzará el análisis de observabilidad. En particular, si se está trabajando en el rediseño de la instrumentación de una planta (revamp), lo más común es que se ingrese su instrumentación actual como punto de partida para el análisis de observabilidad. Sin embargo, cuando se está realizando el diseño de una nueva planta, la inicialización no resulta obvia. En general, el ingeniero elegirá un conjunto inicial de mediciones según sea su nivel de conocimiento del proceso a instrumentar y su experiencia en el área. Para estos casos, resultaría valioso que el DSS contara con alguna herramienta computacional adicional que asistiera al usuario en la configuración inicial de las mediciones. Al respecto, es importante tener en cuenta que el algoritmo de observabilidad es aplicado varias veces en forma iterativa hasta alcanzar una clasificación satisfactoria. Para casos industriales reales, cada iteración puede insumir mucho

tiempo de procesamiento, razón por la cual sería deseable reducir la cantidad de iteraciones requeridas. Como este número depende fuertemente de la calidad de la configuración inicial de los instrumentos, una alternativa interesante sería diseñar un método que obtenga una selección inicial de instrumentos bastante cercana a la configuración ideal, siendo la configuración “ideal” aquella que nos brinda mayor grado de conocimiento del estado de la planta con el menor número de mediciones posible.

Una técnica para abordar este problema puede ser el diseño de algoritmos genéticos con una función de fitness que evalúe las bondades de una solución en términos de las metas del análisis de observabilidad. Obviamente, esto constituye un gran tema de investigación que puede conducir incluso al diseño óptimo de localización de sensores. La incorporación de un algoritmo genético de tales características al DSS desarrollado enriquecería substancialmente este paquete de software.

Otro aspecto importante a considerar como trabajo futuro es la evolución de nuestro actual DSS hacia un paquete completo para diseño de instrumentación. Esto implica la incorporación de un módulo para el análisis de redundancias y otro para la reconciliación de datos, etapa en la cual se ajustan los valores de las mediciones. En la actualidad, las técnicas existentes para la clasificación de variables redundantes no son suficientemente robustas para tratar problemas complejos de gran escala de manera eficiente y además no pueden manejar modelos matemáticos rigurosos que involucran ecuaciones algebraicas fuertemente no lineales. Por tal motivo, resulta necesario el desarrollo de nuevas metodologías para análisis de redundancias, preferiblemente de naturaleza estructural, que permitan extender el DSS sin introducir limitaciones en su rango de aplicabilidad actual. En tal sentido, nuestro grupo de trabajo recientemente ha empezado a investigar este tema.

Finalmente, desde un punto de vista más general, las técnicas de reordenamiento estructural de matrices desarrolladas en esta tesis pueden resultar de interés en otros campos de aplicación fuera de la ingeniería de procesos. En términos matemáticos, los algoritmos de observabilidad propuestos realizan básicamente una descomposición de sistemas de ecuaciones algebraicas generales (lineales o no lineales, singulares o no). Por lo tanto, dado el amplio alcances de las metodologías, es factible encontrar otras nuevas aplicaciones en donde permutar matrices a formar triangular inferior en bloques puede resultar valioso. En definitiva, sería interesante estudiar las posibilidades de extender el rango de aplicabilidad de los métodos diseñados, lo cual aumentaría considerablemente el impacto de las investigaciones y resultados descritos en esta tesis.

## APÉNDICE: MODELADO DE PLANTAS QUÍMICAS USANDO EL DSS

En este apéndice se presentan las interfaces correspondientes a las diferentes categorías de equipos junto con las ecuaciones que describen el estado estacionario de una planta química.

La nomenclatura utilizada se describe en la siguiente tabla:

F	Caudal másico
F <sub>v</sub>	Caudal volumétrico
H	Entalpía
h <sub>L</sub>	Entalpía del líquido saturado
h <sub>v</sub>	Entalpía del vapor saturado
K	Constante de equilibrio
F	Funcionalidad
P	Presión
P <sub>e</sub>	Presión externa
Q	Transferencia de calor
T	Temperatura
T <sub>e</sub>	Temperatura externa
W	Trabajo
S	Flujo de servicio
X	Fracción molar líquido
x <sub>v</sub>	Fracción vaporizada
Y	Fracción molar vapor
Z	Fracción molar líquido ó vapor
	Densidad
	Extensión de la reacción
ρ <sub>l</sub>	Densidad líquido saturado
ρ <sub>v</sub>	Densidad vapor saturado
N <sub>c</sub>	Número total de componentes de una corriente.
N <sub>i</sub>	Número total de corrientes de entrada de un equipo
N <sub>o</sub>	Número total de corrientes de salida de un equipo
N <sub>q</sub>	Número total de corrientes de transferencia de calor.
N <sub>r</sub>	Número total de reacciones.
N <sub>s</sub>	Número total de corrientes de un equipo
N <sub>sc</sub>	Número total de corrientes calientes.
N <sub>sf</sub>	Número total de corrientes frías.
N <sub>w</sub>	Número total de corrientes de transferencia de trabajo.
I	Índice de corrientes
J	Índice de componentes
K	Índice de corrientes
L	Índice asociado al trabajo
M	Índice asociado al calor
I	Corriente de entrada
IC	Corriente de entrada caliente
IF	Corriente de entrada fría
L	Corriente de fase vapor
O	Corriente de salida
OC	Corriente de salida caliente
OF	Corriente de salida fría
V	Corriente de fase líquida

## ECUACIONES DE CORRIENTES

### Ecuaciones de Normalización

Para cada corriente con más de un componente se agrega:

$$\sum_{j=1}^{nc} x_j - 1 = 0 \quad \sum_{j=1}^{nc} y_j - 1 = 0$$

### Ecuaciones de Densidad

Se generan ecuaciones de densidad para aquellas corrientes donde no se ha medido el caudal másico, pero se mide el caudal volumétrico:  $F = Fv \cdot \rho$ .

Para compuestos puros:

$$\delta - f(P, T) = 0$$

Para mezclas:

$$\delta - f(P, T, x, y) = 0$$

Para corrientes que contienen un compuesto puro parcialmente vaporizado:

$$\delta - x_v \delta_v + (1 - x_v) \delta_L = 0$$

$$\delta_v - f(T) = 0$$

$$\delta_L - f(T) = 0$$

$$T - f(P) = 0$$

$$\begin{cases} x_v - f(P_e, T_e, P) = 0, & \text{si } x_v \text{ es medida} \\ x_v - f(h, P) = 0, & \text{si } x_v \text{ es no medida} \end{cases}$$

## Ecuaciones de Entalpía

Se generan ecuaciones de entalpía para todas las corrientes que están involucradas en balances de energía.

Para compuestos puros:

$$h - f(P, T) = 0$$

Para mezclas:

$$h - f(P, T, x, y) = 0$$

Para corrientes que contienen un compuesto puro parcialmente vaporizado:

$$h - x_v h_v + (1 - x_v) h_L = 0$$

$$h_v - f(T) = 0$$

$$h_L - f(T) = 0$$

$$T - f(P) = 0$$

$$\begin{cases} x_v - f(P_e, T_e, P) = 0, & \text{si } x_v \text{ es medida} \\ x_v - f(h, P) = 0, & \text{si } x_v \text{ es no medida} \end{cases}$$

## ECUACIONES DE EQUIPOS

En el DSS se han considerado los siguientes equipos de proceso:

Equipos con cambio de composición	Equipos sin cambio de composición
<ul style="list-style-type: none"><li>• Mezclador de distinta entalpía - Distinta composición</li><li>• Columna de destilación</li><li>• Separador</li><li>• Separador isotérmico</li><li>• Condensador</li><li>• Rebullidor</li><li>• Reactor</li><li>• Equipo general</li></ul>	<ul style="list-style-type: none"><li>• Mezclador de distinta entalpía - Igual composición</li><li>• Mezclador de igual entalpía</li><li>• Divisor</li><li>• Bomba</li><li>• Bomba isotérmica</li><li>• Compresor</li><li>• Aeroenfriador</li><li>• Turbina</li><li>• Intercambiador de calor sin cambio parcial de fase</li></ul>

Las ecuaciones de balance correspondientes a cada uno de los equipos previamente citados se detallan a continuación.

## EQUIPO GENERAL

Es una unidad donde no se produce reacción química ni transferencia de calor y/o trabajo.

**Nuevo Equipo General**

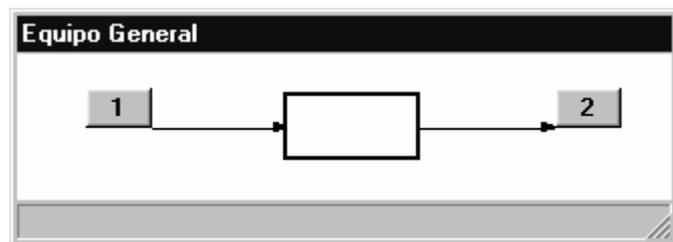
Nombre:

Entradas: 1

Salidas: 1

Entradas FEP: 0

Salidas FEP: 0



### Ecuación de Balance de Masa Global

$$\sum_{i=1}^{ni} F_i^I - \sum_{i=1}^{no} F_i^O = 0$$

### Ecuación de Balance de Masa por Componente

$$\sum_{i=1}^{ni} F_i^I z_{j,i}^I - \sum_{i=1}^{no} F_i^O z_{j,i}^O = 0 \quad j: 1 \dots nc$$

### Ecuación de Balance de Energía

$$\sum_{i=1}^{ni} F_i^I h_i^I - \sum_{i=1}^{no} F_i^O h_i^O = 0$$

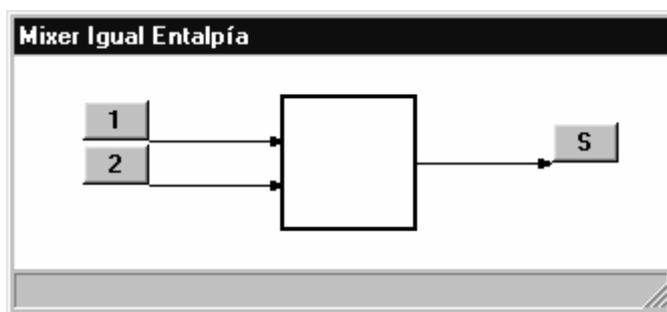
Es una unidad donde varias corrientes se unifican.

## MEZCLADORES

**Nuevo Mixer Igual Entalpía**

Nombre:

Entradas:



### **Mezclador de Igual Entalpía**

Es un mezclador cuyas corrientes de entrada poseen igual presión, temperatura y composición.

#### *Ecuación de Balance de Masa Global*

$$\sum_{i=1}^{ni} F_i^I - F^O = 0$$

#### *Ecuación de Balance de Masa por Componente*

$$z_{j,i} - z_{j,k} = 0 \quad j: 1 \dots nc, \quad i: 1 \dots (ns-1), \quad k: (i+1) \dots ns$$

#### *Ecuación de Balance de Energía*

$$\begin{aligned} h_i - h_k &= 0 & i: 1 \dots (ns-1), \quad k: (i+1) \dots ns \\ P_i - P_k &= 0 \\ T_i - T_k &= 0 \end{aligned}$$

### **Mezclador de Distinta Entalpía / Igual Composición**

Es un mezclador cuyas corrientes de entrada poseen distinta presión y/o temperatura e igual composición.

#### *Ecuación de Balance de Masa Global*

Error! Not a valid link.

### ***Ecuación de Balance de Masa por Componente***

Error! Not a valid link.

### ***Ecuación de Balance de Energía***

$$\sum_{i=1}^{ni} F_i^I h_i^I - F^O h^O = 0$$

### **Mezclador de Distinta entalpía / Distinta Composición**

Es un mezclador cuyas corrientes de entrada poseen distinta presión, temperatura y composición.

### ***Ecuación de Balance de Masa Global***

Error! Not a valid link.

### ***Ecuación de Balance de Masa por Componente***

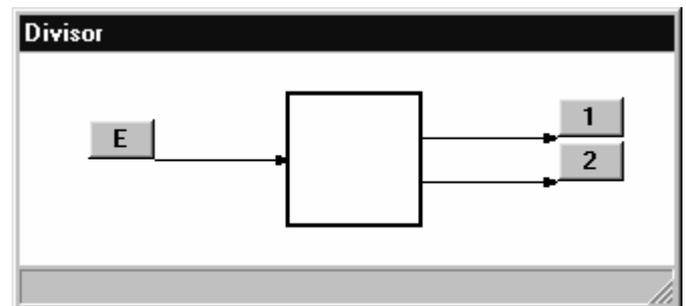
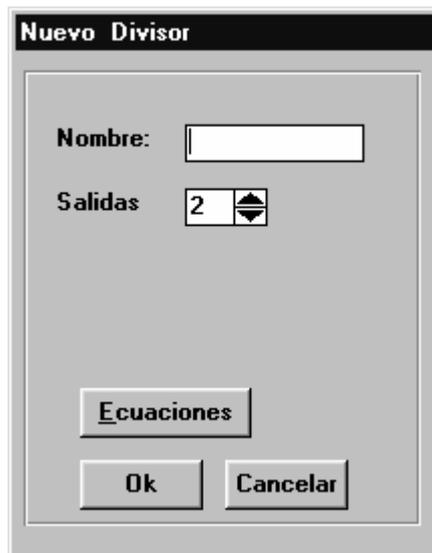
$$\sum_{i=1}^{ni} F_i^I z_{j,i}^I - F^O z_j^O = 0$$

### ***Ecuación de Balance de Energía***

Error! Not a valid link.

## DIVISOR

Es una unidad que subdivide una corriente de entrada.



### Ecuación de Balance de Masa Global

$$F^I - \sum_{i=1}^{no} F_i^O = 0$$

### Ecuación de Balance de Masa por Componente

$$z_j^I - z_{j,i}^O = 0 \quad i: 1 \dots no, j: 1 \dots nc$$

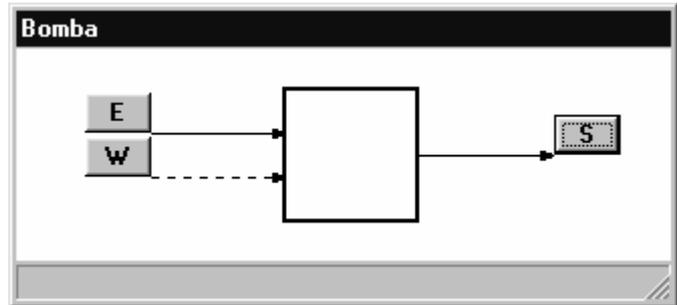
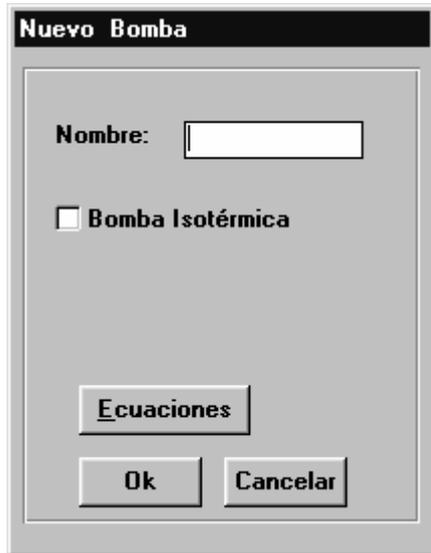
$$z_{j,i} - z_{j,k} = 0 \quad i: 1 \dots (ns - 1), k: (i + 1) \dots ns$$

### Ecuación de Balance de Energía

Error! Not a valid link.

## BOMBA

Es una unidad donde una corriente líquida recibe trabajo.



### Ecuación de Balance de Masa Global

$$F^I - F^O = 0$$

### Ecuación de Balance de Masa por Componente

$$x_j^I - x_j^O = 0 \quad j: 1 \dots nc$$

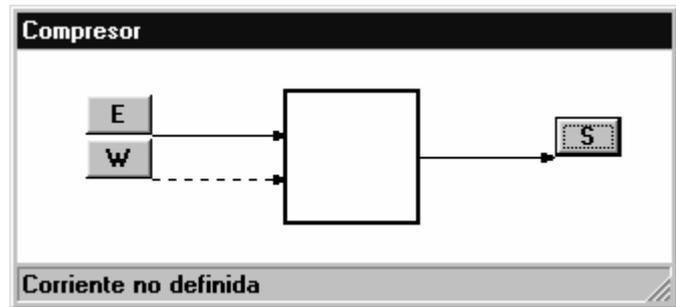
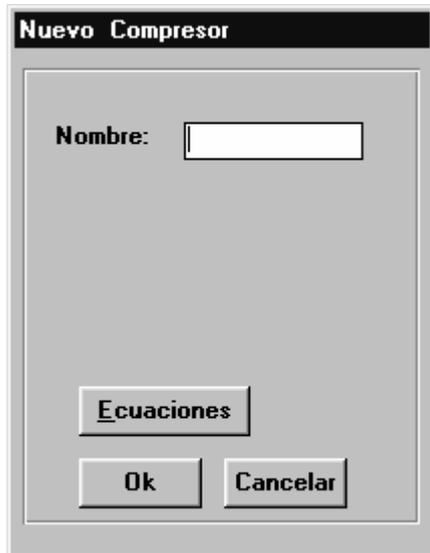
### Ecuación de Balance de Energía

$$F^I h^I - F^O h^O + W = 0$$

Si se asume que la bomba es isotérmica, se agrega la siguiente ecuación:  $T^I - T^O = 0$

## COMPRESOR

Es una unidad donde se comprime un gas.



### Ecuación de Balance de Masa Global

Error! Not a valid link.

### Ecuación de Balance de Masa por Componente

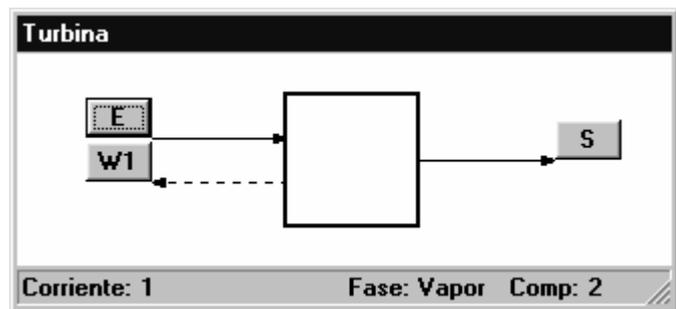
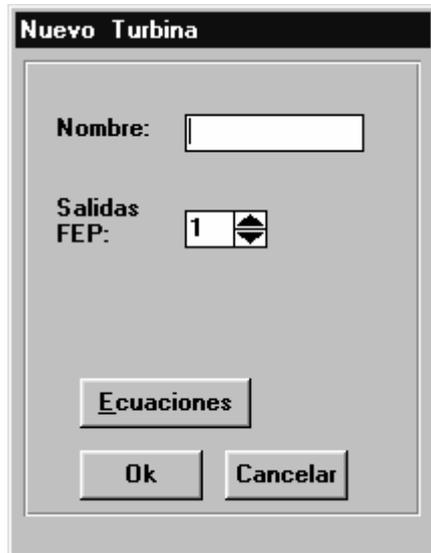
$$y_j^I - y_j^O = 0 \quad j: 1..nc$$

### Ecuación de Balance de Energía

Error! Not a valid link.

## TURBINA

Es una unidad que descomprime un gas generando trabajo.



### Ecuación de Balance de Masa Global

Error! Not a valid link.

### Ecuación de Balance de Masa por Componente

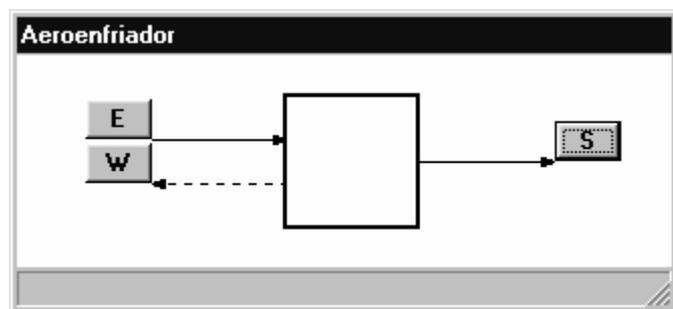
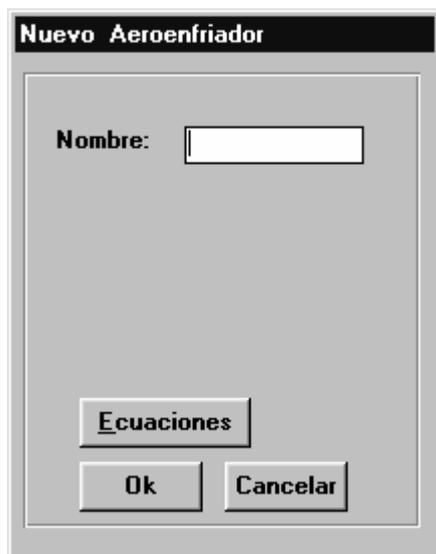
$$y_j^I - y_j^O = 0 \quad j: 1 \dots nc$$

### Ecuación de Balance de Energía

$$F^I h^I - F^O h^O - \sum_{l=1}^{mw} W_l = 0$$

## AEROENFRIADOR

Es un intercambiador de calor donde el fluido de proceso reduce su temperatura por enfriamiento por aire.



### Ecuación de Balance de Masa Global

Error! Not a valid link.

### Ecuación de Balance de Masa por Componente

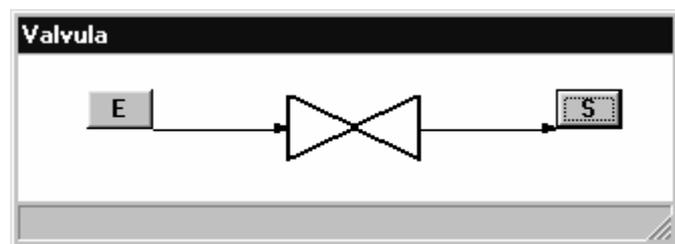
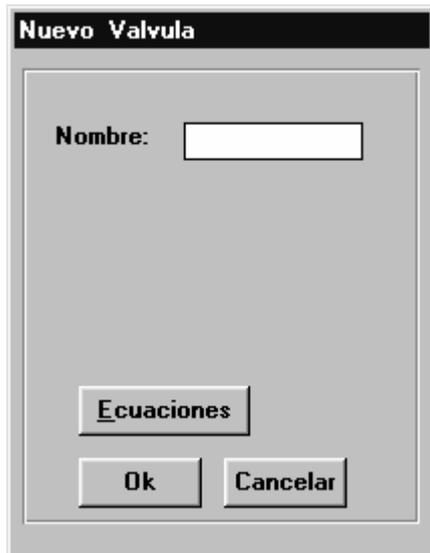
$$z_j^I - z_j^O = 0 \quad j: 1 \dots nc$$

### Ecuación de Balance de Energía

$$F^I h^I - F^O h^O - Q = 0$$

## VÁLVULA JOULE-THOMPSON

Es un accesorio que ofrece resistencia al flujo, reduciendo la presión sin producir trabajo y cuyos cambios globales en energía cinética y potencial son despreciables.



### Ecuación de Balance de Masa Global

Error! Not a valid link.

### Ecuación de Balance de Masa por Componente

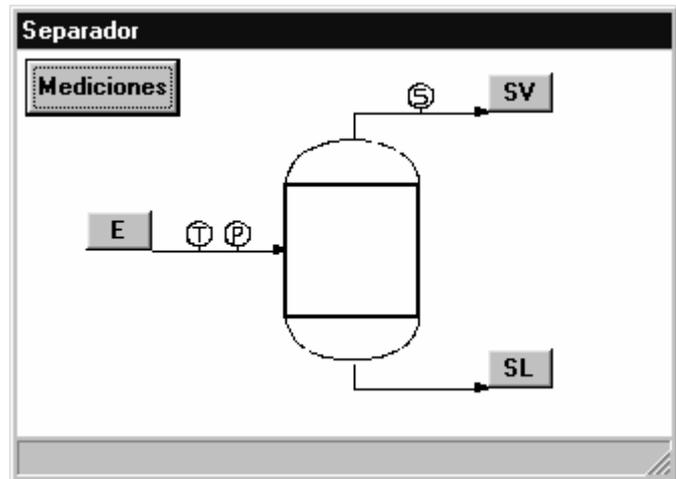
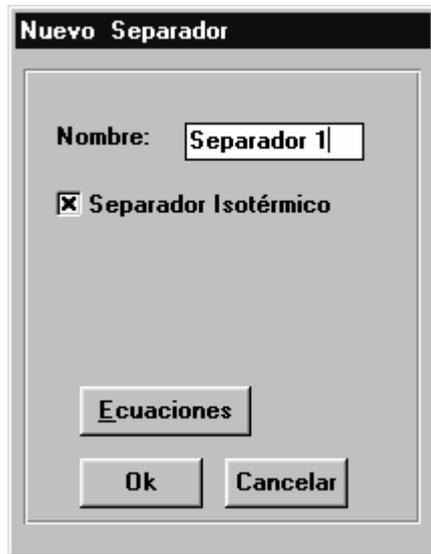
$$z_j^I - z_j^O = 0 \quad j: 1 \dots nc$$

### Ecuación de Balance de Energía

$$h^I - h^O = 0$$

## SEPARADORES

El separador es una unidad donde la alimentación se separa en una fase líquida y otra vapor en equilibrio.



**Alimentación: un componente puro**

**Ecuación de Balance de Masa Global**

$$F^I - F^V - F^L = 0$$

**Ecuación de Balance de Energía**

$$F^I h^I - F^V h^V - F^L h^L = 0$$

$$x_v F^I - F^V = 0$$

$$P - f(T) = 0$$

$$\begin{aligned} P - P^V &= 0 & T - T^V &= 0 \\ P - P^L &= 0 & T - T^L &= 0 \\ P^L - P^V &= 0 \end{aligned}$$

Para el caso de un separador de un componente isotérmico se agregan las ecuaciones:

$$\begin{aligned} T^I - T^V &= 0 \\ T^I - T^L &= 0 \\ T^I - T &= 0 \end{aligned}$$

### **Alimentación: mezcla multicomponente**

#### **Ecuación de Balance de Masa Global**

Error! Not a valid link.

#### **Ecuación de Balance de Masa por Componente**

$$F^I z_j^I - F^V y_j^V - F^L x_j^L = 0 \quad j: 1 \dots nc$$

$$y_j - K_j x_j = 0 \quad j: 1 \dots nc \quad K_j - f(P, T, x_j, y_j) = 0 \quad j: 1 \dots nc$$

#### **Ecuación de Balance de Energía**

Error! Not a valid link.

$$\begin{aligned} T - T^V &= 0 & P - P^V &= 0 \\ T - T^L &= 0 & P - P^L &= 0 \\ T^L - T^V &= 0 & P^L - P^V &= 0 \end{aligned}$$

Para el caso de un separador de multicomponentes isotérmico se agregan las ecuaciones:

$$T^I - T^V = 0 \quad P^I - P^V = 0$$

$$T^I - T^L = 0 \quad P^I - P^L = 0$$

$$T^I - T = 0 \quad P^I - P = 0$$

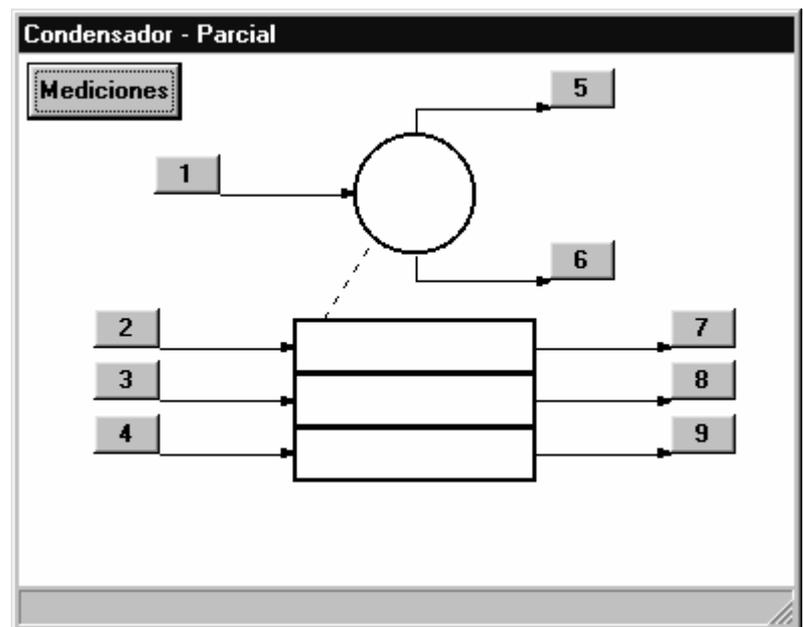
## CONDENSADOR

Es una unidad de intercambio de calor en la cual se produce la condensación parcial ó total de la fase de entrada.

**Nuevo Condensador - Parcial**

Nombre:

Entradas Frías:



### Ecuación de Balance de Masa Global

Error! Not a valid link.

$$S_m^I - S_m^O = 0 \quad m: 1 \dots nq$$

### **Ecuación de Balance de Masa por Componente**

$$F^I y_j^I - F^V y_j^V - F^L x_j^L = 0 \quad j: 1 \dots nc$$

Para el caso de condensación parcial:

$$y_j - K_j x_j = 0 \quad j: 1 \dots nc$$

$$K_j - f(P, T, x_j, y_j) = 0 \quad j: 1 \dots nc$$

Para el caso de condensación total:

$$y_{j,m}^I - x_{j,m}^O = 0 \quad m: 1 \dots nq, j: 1 \dots nc$$

### **Ecuación de Balance de Energía**

$$F^I h^I - F^V h^V - F^L h^L - \sum_{m=1}^{nq} Q_m = 0$$

$$S_i^I h_i^I - S_i^O h_i^O + Q_m = 0 \quad m: 1 \dots nq$$

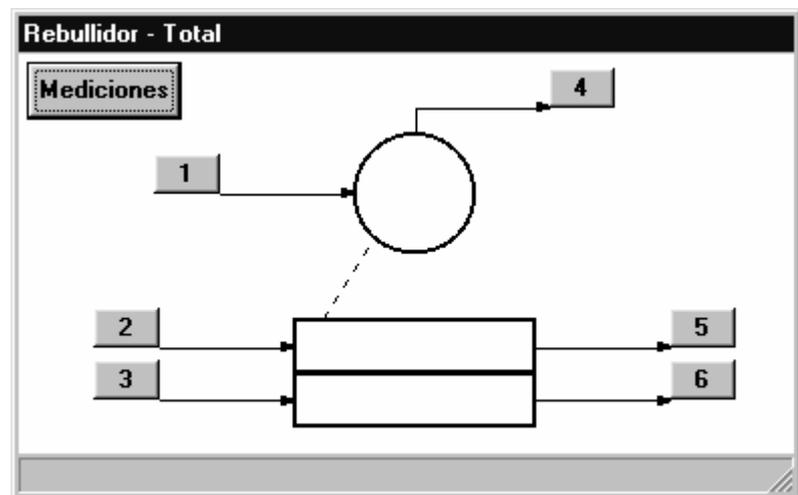
## REBULLIDOR

Es una unidad de intercambio de calor en la que se vaporiza total ó parcialmente la corriente de entrada.

**Nuevo Rebullidor - Total**

Nombre:

Entradas Frías:



### Ecuación de Balance de Masa Global

$$F^I - F^V - F^L = 0$$

Error! Not a valid link.

### Ecuación de Balance de Masa por Componente

Error! Not a valid link.

Para el caso de vaporización parcial:

Error! Not a valid link.

Error! Not a valid link.

Para el caso de vaporización total:

$$x_{j,m}^I - y_{j,m}^O = 0 \quad m: 1 \dots nq, j: 1 \dots nc$$

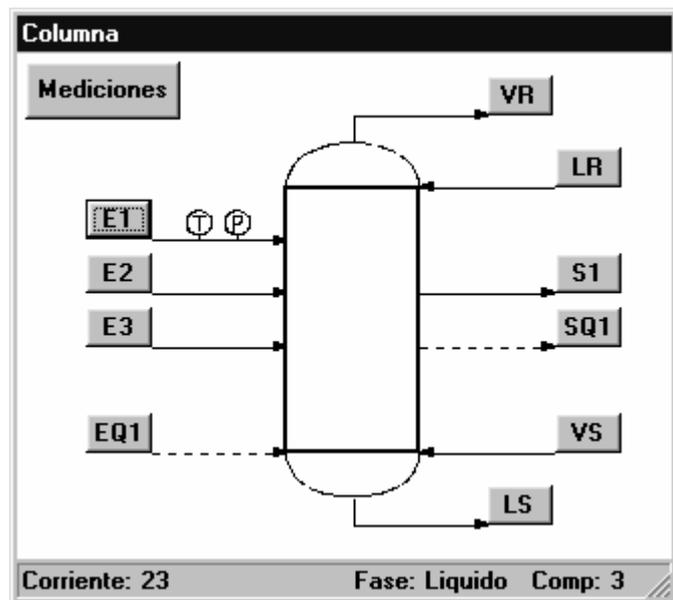
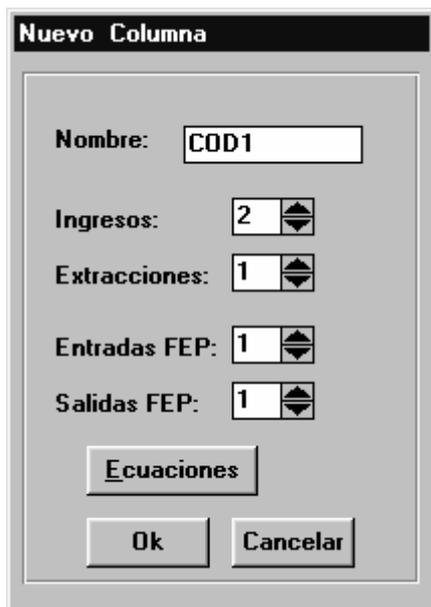
### Ecuación de Balance de Energía

$$F^I h^I - F^V h^V - F^L h^L + \sum_{m=1}^{nq} Q_m = 0$$

$$S_i^I h_i^I - S_i^O h_i^O - Q_m = 0 \quad m: 1 \dots nq$$

## COLUMNA DE DESTILACIÓN

Es una unidad que produce una separación en base a la creación de dos fases líquido y vapor coexistentes con concentraciones distintas de cada especie.



## Ecuación de Balance de Masa Global

Error! Not a valid link.

## Ecuación de Balance de Masa por Componente

Error! Not a valid link.

## Ecuación de Balance de Energía

$$\sum_{i=1}^{ni} F_i^I h_i^I - \sum_{i=1}^{no} F_i^O h_i^O + \sum_{m=1}^{nqi} Q_m^I - \sum_{m=1}^{nqo} Q_m^O = 0$$

## REACTOR

Es una unidad en la cual se producen reacciones químicas con el objeto de obtener productos de interés.

**Nuevo Reactor**

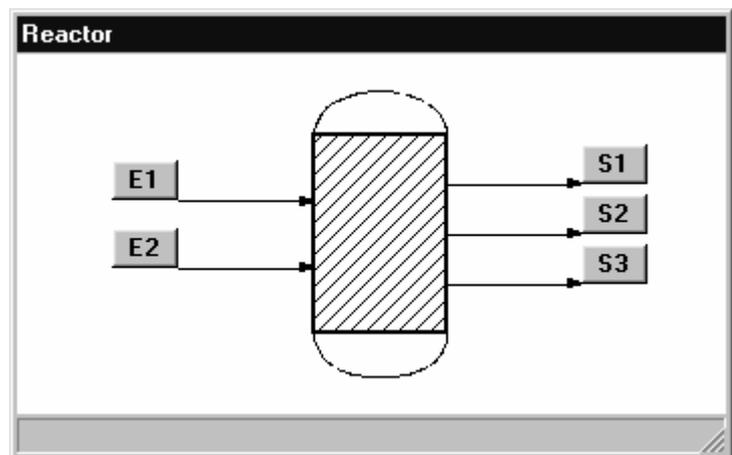
Nombre:

Entradas:

Salidas:

Tipo:

Nro. Reacciones:



### Ecuación de Balance de Masa Global

$$F^I - F^O = 0$$

### Ecuación de Balance de Masa por Componente

$$F^I z_j^I - F^O z_j^O + \sum_{k=1}^{nr} \xi_k = 0 \quad j: 1 \dots nc$$

### Ecuación de Balance de Energía

$$\begin{cases} F^I h^I - F^O h^O + \sum_{i=1}^{nr} Qr_i = 0 & \text{reactor adiabático} \\ F^I h^I - F^O h^O - Q + \sum_{i=1}^{nr} Qr_i = 0 & \text{reactor endotérmico} \\ F^I h^I - F^O h^O + Q + \sum_{i=1}^{nr} Qr_i = 0 & \text{reactor exotérmico} \end{cases}$$

$$Qr = \frac{\xi_{rA}}{|v_A|} \left[ \sum_{j=1}^{npr} v_j (h_{j(T_o)}^r - h_{j(T_r)}^0) + |v_A| \Delta H_r^0 \right]$$

$$\Delta H_r^0 = \frac{1}{|v_A|} \left[ \sum_{k=1}^{nr} v_k (h_{k(T_r)}^0 - h_{k(25^\circ)}^{25^\circ}) + |v_A| \Delta H_{r(25^\circ)}^{25^\circ} + \sum_{j=1}^{npr} v_j (h_{j(T_r)}^0 - h_{j(25^\circ)}^{25^\circ}) \right]$$

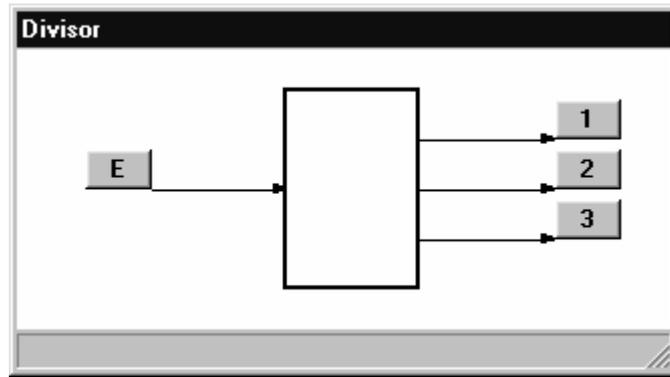
### DIVISOR

Es una unidad de transferencia de calor donde los fluidos ingresantes no sufren cambios de fase.

**Nuevo Divisor**

Nombre:

Salidas  



### Ecuación de Balance de Masa Global

$$F_i^{IC} - F_i^{OC} = 0 \quad i: 1 \dots nsc$$

$$F_i^{IF} - F_i^{OF} = 0 \quad i: 1 \dots nsf$$

### Ecuación de Balance de Masa por Componente

$$z_{j,i}^{IC} - z_{j,i}^{OC} = 0 \quad i: 1 \dots nsc, j: 1 \dots nc$$

$$z_{j,i}^{IF} - z_{j,i}^{OF} = 0 \quad i: 1 \dots nsf, j: 1 \dots nc$$

### Ecuación de Balance de Energía

$$F_i^{IC} h_i^{IC} - F_i^{OC} h_i^{OC} - \sum_{k=1}^{nsf} Q_k = 0 \quad i: 1 \dots nsc$$

$$F_i^{IF} h_i^{IF} - F_i^{OF} h_i^{OF} + \sum_{k=1}^{nsc} Q_k = 0 \quad i: 1 \dots nsf$$

## REFERENCIAS

-  **Aho, A.V.; Hopcroft, J.E. y Ullman, J.D.** *Data Structures and Algorithms*, capítulos 2 y 6, Addison-Wesley Publishing Company, **1983**.
-  **Anderson, T.E.; Culler, D.E. y Patterson, D.** "A Case for NOW (Networks of Workstations)", *IEEE Micro*, 15, 54-64, **1995**.
-  **ASPEN Project.** "Functional Specifications for ASPEN", *Dept. of Chem. Eng., M.I.T.*, (<http://www.aspentech.com>) **1977**.
-  **Asratian, A.S.; Denley, T.M.J. y Häggkvist, R.** *Bipartite Graphs and their Applications*, Cambridge University Press: Cambridge (RU), **1998**.
-  **Bellman, R.; Cooke, K.L. y Lockett J.A.** *Algorithms Graphs and Computers*, Academic Press: London, (RU), **1970**.
-  **Bennett, J.L.** *Building Decision Support Systems*, Addison-Wesley, **1983**.
-  **Berge, C.** *Hypergraphes. Combinatoires des ensembles finis*, Gauthier-Villars: Paris (Francia), **1987**.
-  **Berge, C.**, "Hypergraphs", *Graph Theory. 3, capítulo 9*, Academic Press, **1988**.
-  **Biegler, L.T.**, "Chemical Process Simulation", *Chem. Eng. Prog.*, 85, 10, 50-61, **1989**.
-  **Biegler, L.T.; Grossman, I.E. y Westerberg, A.W.** "Systematic Methods of Chemical Process Design", Prentice Hall, **1997**.
-  **Bike, S.** "Design of an Ammonia Synthesis Plant", *CACHE Case Study*, Dept. Chem. Engng, Carnegie Mellon University, **1985**.

- 📖 **Bugaeva, L.N.; Beznosik, Y.; Statjukha, G.A. y Kvitka, A.A.** “An Application of Expert Systems to the Choice, Simulation and Development of Gas-Purification Processes”, *Comp. & Chem. Engng.*, Vol. 20S-A, S401-S406, **1996**.
- 📖 **Bunch, J.R. y Rose D.J.** *Sparse Matrix Computations*, Academic Press: London (RU), **1976**.
- 📖 **Christofides, N.** *Graph Theory. An Algorithmic Approach*, Academic Press: London (RU), **1975**.
- 📖 **Crowe, C.M.** “Reconciliation of Process Flow Rates by Matrix Projection Part II: The Non-linear Case”, *AIChE Jl.*, 32, 616-623, **1986**.
- 📖 **Crowe, C. M.** “Observability and Redundancy of Process Data for Steady State Reconciliation”, *Chem. Eng. Sci.*, 44, 2909-2917, **1989**.
- 📖 **Crowe, C.M.; García Campos, Y.A. y Hrymak A.** “Reconciliation of Process Flow Rates by Matrix Projection Part I: Linear Case”, *AIChE Jl.*, 29, 881-888, **1983**.
- 📖 **Crowl, L.** "How to Measure, Present and Compare Parallel Performance", *Parallel and Distributed Technology*, 2, 9-25, **1994**.
- 📖 **DATACON™**, Simulation Sciences Inc., Fullerton, CA, USA (<http://www.simsci.com>).
- 📖 **Duff, I.S.** “On Algorithms for Obtaining a Maximum Transversal”, *ACM Trans. Math. Softw.*, 7, 315-330, **1981**.
- 📖 **Duff, I.S. y Reid, J.K.** "An Implementation of Tarjan’s Algorithm for the Block Triangularization of a Matrix", *ACM Trans. Math. Softw.*, 4, 137-147, **1978**.
- 📖 **Duff, I.S.; Erisman, A.M. y Reid J.K.** *Direct Methods for Sparse Matrices*, Oxford Science Pub.: Chippenham, Wiltshire (RU), **1997**.

- 
-  **Dulmage, A.L. y Mendelsohn, N.S.** “Coverings of Bipartite Graphs”, *Can. J. Math.*, 10, 517-534, **1958**.
-  **Dulmage, A.L. y Mendelsohn, N.S.** “Two Algorithms for Bipartite Graphs”, *J. Soc. Ind. Appl. Math.*, 11, 183-194, **1963**.
-  **Evans, L.B.; Joseph, B.; Tsang, M. y White, G.** "PLEXSYS II- PLEX Data management and Plex Building Program - User's Manual", Segunda Revisión, Chem. Eng. Dept. and Energy Lab., MIT, A-26, **1977**.
-  **Even, S.** *Graph Algorithm*, Computer Science Press: Maryland (EEUU), **1979**.
-  **Freund, R. y Siegel, H.**, "Heterogeneous Processing", *IEEE Computer*, 26, 13-17, **1993**.
-  **Gibbons, A.** *Algorithmic Graph Theory*, Cambridge University Press: Newcastle (RU), **1994**.
-  **Golumbic, M.C.** *Algorithmic Graph Theory and Perfect Graphs*, Academic Press: New York (EEUU), **1980**.
-  **Gustavson F. G.**, “Finding the Block Lower-Triangular Form of a Sparse Matrix”, Bunch and Rose eds., *Sparse Matrix Computations*, Academic Press: London (RU), **1976**.
-  **Hall, M.** “An Algorithm for Distinct Representatives”, *Am. Math. Monthly*, 63, 716-717, **1956**.
-  **Hartmann, K.; Behrendt, U. y Dietzsch, L. A.** “Computer-Aided Decision Support System for the Optimal Long-Term Planning in Large-Scale Chemical and Related Process Systems”, *Computer Applications in Chemical Engineering*, Elsevier Publishers B.V., Amsterdam, 165-170, **1990**.

-  **Hopcroft, J. E. y Karp, R. M.** "An  $n^{5/2}$  Algorithm for Maximum Matchings in Bipartite Graphs", *SIAM J. Comput.* 2, 4, 225-231, **1973**.
-  **Joris, P. y Kalitventzeff, B.** "Process Measurement Analysis and Validation", *XVIII Congr. The Use of Comp. in Chem. Engng*, CEF'87, Italia, 41-46, Abril de 1987.
-  **Karpinski, M. y Rytter, W.** *Fast Parallel Algorithms for Graph Matching Problems*, Oxford University Press, Oxford, England, **1998**.
-  **Kretsovalis, A. y Mah, R.S.H.** "Observability and Redundancy Classification in Multicomponent Process Networks", *AIChE Jl.*, 33, 70-82, **1987**.
-  **Kretsovalis, A. y Mah, R.S.H.** "Observability and Redundancy Classification in Generalized Process Networks - I. Theorems", *Comp. & Chem. Engng.*, 12, 671-687, **1988a**.
-  **Kretsovalis, A. y Mah, R.S.H.** "Observability and Redundancy Classification in Generalized Process Networks - II. Algorithms", *Comp. & Chem. Engng.* 12, 689-703, **1988b**.
-  **Khokhar, A.A.; Prasanna, V.K.; Shaaban M.E. y Wang. C.** "Heterogeneous Computing: Challenges and Opportunities", *IEEE Computer*, 26, 18-27, **1993**.
-  **Leone, H.** "A Knowledge-Based System for HAZOP Studies", *Comp. & Chem. Engng.*, Vol. 20S-A, S369-S374, **1996**.
-  **Lygeros, A.; Maroulis, Z.B. y Prokopakis, G.J.** "An Integrated Software Package for total Site Integration and Utilities Design", *Comp. & Chem. Engng.*, Vol. 20S-B, S1607-S1612, **1996**.

- 
- 📖 **Madron, F.** *Process Plant Performance. Measurement and Data Processing for Optimization and Retrofits*, Ellis Horwood Ltd.: Chichester, England, **1992**.
- 📖 **Meyer, M.; Koehret, B. y Enjalbert, M.** "Data Reconciliation on Multicomponent Network Process", *Comp. & Chem. Engng.*, 17, 807-817, **1993**.
- 📖 **Motard, R.L.; Lee, H.M.; Barkley, R.W. y Ingels D.M.** "CHESS: Chemical Engineering Simulation System - System Guide", Tech Publishing Co., Universidad de Houston, Texas, USA, **1968**.
- 📖 **Pissanetzky, S.** *Sparse Matrix Technology*, Academic Press Inc: London, (RU), **1984**.
- 📖 **Ponzoni, I.; Sánchez, M.C. y Brignole, N.B.** "Algoritmos Eficientes en Problemas de Asignación", *Mecánica Computacional*, Vol. XV, 415-424., **1995**.
- 📖 **Ponzoni, I.; Sánchez, M.C. y Brignole, N.B.** "An Efficient Implementation of the First Least-Connected Node Algorithm", *Mecánica Computacional*, Vol. XVIII, 629-638, **1997**.
- 📖 **Ponzoni, I.; Sánchez, M.C. y Brignole, N.B.** "CDHG: a New Partitioning Algorithm based on the Detection of Cycles in Hypergraphs", *Latin American Applied Research*, Vol. 28, N°1/2, 31-36, **1998a**.
- 📖 **Ponzoni, I.; Sánchez, M.C. y Brignole, N.B.** "Permutation of Sparse Matrices to a specific lower BTF using Graph Decompositions", *Electronical Journal of the Argentine Society for Informatics and Operations Research*, 1, 1, 76-87, **1998b**. Link: <http://www.dc.uba.ar/sadio/ejs>
- 📖 **Ponzoni, I.; Sánchez, M.C. y Brignole, N.B.** "A New Hybrid Approach for Instrumentation Design of Chemical Plants", *IV WCCM (4th World Congress on*

- Computational Mechanics*), publicado en el CD-ROM del congreso. Buenos Aires (Argentina). 29 junio-2 de julio, **1998c**.
- 📖 **Ponzoni, I.; Sánchez, M.C. y Brignole, N.B.** "A Graph-Based Algorithm for Constrained Partitioning of Sparse Matrices", *IV WCCM (4th World Congress on Computational Mechanics)*, publicado en el CD-ROM del congreso. Buenos Aires (Argentina). 29 junio-2 de julio, **1998d**.
- 📖 **Ponzoni, I.; Sánchez, M.C. y Brignole, N.B.** "A New Structural Algorithm for Observability Classification", *Ind. Eng. Chem. Res.*, 38, 8, 3027-3035, **1999**.
- 📖 **Ponzoni, I.; Vazquez, G.E.; Sánchez, M.C. y Brignole, N.B.** "A Computer-Aided DSS for Observability Analysis", *Signal Processing, Communications and Computer Science*, World Scientific and Engineering Society Press, Electrical and Computer Engineering Series, 222-227, **2000**.
- 📖 **Ponzoni, I.; Vazquez, G.E.; Sánchez, M.C. y Brignole, N.B.** "Parallel Observability Analysis on Networks of Workstations", *Comp. & Chem. Engng.*, 25, 7-8, 997-1002, **2001**.
- 📖 **Pothen, A. y Fan, C-J.** "Computing the Block-Triangular Form of a Sparse Matrix", *ACM Trans. Math. Softw.*, 16, 4, 303-324, **1990**.
- 📖 **PROCESS®**, Process Simulation International, Simulation Sciences Inc., Fullerton, CA, USA (<http://www.simsci.com>).
- 📖 **Romagnoli, J.A. y Sánchez, M.C.** *Data Processing and Reconciliation for Chemical Process Operations*, Academic Press Inc: San Diego (EEUU), **1999**.
- 📖 **Romagnoli, J.A. y Stephanopoulos, G.** "On the Rectification of Measurement Errors for Complex Chemical Plants", *Chem. Engng. Sci.*, 35, 1067-1081, **1980**.

- 
-  **Rosen, E.M. y Pauls, A.C.** "Computer-Aided Chemical Process Design: The FLOWTRAN System", *Comp. Chem. Eng.* 1,1, **1977**.
-  **Sánchez, M.C.; Bandoni, A.J. y Romagnoli; J.A.** "PLADAT: A Package for Process Variable Classification and Plant Data Reconciliation", *Comp. & Chem. Engng.*, S499-S506, **1992**.
-  **SIGMAFINE®**, OSI Software Inc., San Leandro, CA, USA (<http://www.osisoft.com>).
-  **Stadtherr, M.A.; Gifford, W.A. y Scriven, L.E.** "Efficient Solution of Sparse Sets of Design Equations", *Chem. Engng. Sci.*, 29, 1025-1034, **1974**.
-  **Tarjan, R. E.** "Depth-First Search and Linear Graph Algorithms", *SIAM J. Comput.*, 1, 2, 146-160, **1972**.
-  **Tsai, C.S. y Chang, C.T.** Dynamic Process Diagnosis via Integrated Neural Networks, *Comp. & Chem. Engng.*, Vol. 19S, S747-S752, **1995**.
-  **Vaclavek, V.** "Studies on System Engineering. III: Optimal Choice of the Balance Measurements in Complicated Chemical Systems", *Chem. Engng. Sci.*, 24 (6), 947-955, **1969**.
-  **Vaclavek, V. y Loucka, M.** "Selection of Measurements Necessary to Achieve Multicomponent Mass Balances in Chemical Plants", *Chem. Engng. Sci.*, 31 (12), 1199-1205, **1976**.
-  **VALI®**, Process Data Reconciliation in Refineries, Petrochemical and Chemical Plants, Belsim S.A., Liege, Bélgica (<http://www.belsim.com>).

-  **Vazquez, G.E.; Ponzoni, I.; Sánchez, M.C. y Brignole N.B.** "ModGen: A Model Generator for Instrumentation Analysis", *Advances in Engineering Software. Including Computing Systems in Engineering.*, 32, 1, 37-48, **2001**.

## INDICE DE ALGORITMOS

<i>Algoritmo 2.1. DFS sobre grafos (DFS).</i>	23
<i>Algoritmo 2.2. DFS transversal sobre grafos (DFST).</i>	23
<i>Algoritmo 2.3. DFS para detección de componentes fuertes de digrafos (DFSCCD).</i>	28
<i>Algoritmo 2.4. Detección de todas las componentes fuertes de un digrafo (TCCD)</i>	28
<i>Algoritmo 2.5. Pareamiento maximal de un bigrafo (PMB).</i>	31
<i>Algoritmo 4.1. Estrategia global para detección de subconjuntos de asignación (GS-FLCN).</i>	57
<i>Algoritmo 4.2. Subrutina 2.</i>	60
<i>Algoritmo 4.3. Algoritmo Modificado</i>	61
<i>Algoritmo 4.4. First Least-Connected Node (FLCN).</i>	64
<i>Algoritmo 5.1. Detección de Ciclos en Hipergrafos (CDHG).</i>	85
<i>Algoritmo 5.2. Función EncuentraAutoAristas.</i>	87
<i>Algoritmo 5.3. Función BuscarHGP.</i>	88
<i>Algoritmo 5.4. Función TestCiclo.</i>	90
<i>Algoritmo 5.5. Enfoque Híbrido (EH).</i>	96
<i>Algoritmo 6.1. Método Directo para Clasificación de Variables.</i>	104

## INDICE DE FIGURAS

<i>Figura 2.1. Representación gráfica de un grafo no dirigido <math>G</math>.</i>	19
<i>Figura 2.2. Representación gráfica de un digrafo <math>G</math>.</i>	21
<i>Figura 2.3. Componentes fuertes detectadas por TCCD.</i>	29
<i>Figura 2.4. Representación gráfica de un bigrafo <math>G</math>.</i>	30
<i>Figura 2.5. Matrices de incidencia.</i>	33
<i>Figura 2.6. Obtención de la FTB aplicando los algoritmos PMB y TCCD.</i>	36
<i>Figura 2.7. FTB superior obtenida usando la clasificación de Dulmage-Mendelsohn</i>	38
<i>Figura 2.8. Un hipergrafo y su matriz de incidencia</i>	39
<i>Figura 3.1. Diagrama de flujo de un proceso y su grafo <math>G</math> subyacente.</i>	47
<i>Figura 3.2. Matriz de ocurrencia reordenada obtenida por Romagnoli y Stephanopoulos.</i>	52
<i>Figura 4.1. Submatriz <math>S</math> y digrafo <math>G</math> asociado a <math>S^T S</math>.</i>	63
<i>Figura 4.2. Subconjunto del ejemplo de Pissanetzky detectado por GS-FLCN.</i>	67
<i>Figura 4.3. Diagrama de flujo del proceso (Joris y Kalitventzeff, 1986).</i>	68
<i>Figura 4.4. Esquema simplificado de la planta de producción de etano.</i>	70
<i>Figura 4.5. Subconjunto de asignación de tamaño 5.</i>	73
<i>Figura 4.6. Grupo de intercambiadores de calor en el área criogénica.</i>	74
<i>Figura 4.7. Subconjunto de asignación del ciclo S1-S9 de la figura 3.6.</i>	74
<i>Figura 4.8. Estructura General de GS-FLCN y sección paralelizada.</i>	77
<i>Figura 4.9. Descomposición del espacio de búsqueda.</i>	77
<i>Figura 4.10. Diagrama de flujo de la planta de síntesis de amoníaco.</i>	78
<i>Figura 6.1. Comparación de FTBs.</i>	102
<i>Figura 6.2. Esquema general del método directo</i>	103
<i>Figura 6.3. Clasificación de nodos y descomposición gruesa</i>	107

<i>Figura 6.4. Descomposición Gruesa.</i>	108
<i>Figura 6.5. Descomposición Fina de <math>N_2 = (SC2, SR2)</math>.</i>	109
<i>Figura 6.6. Reasignación del bloque <math>N_1</math>.</i>	113
<i>Figura 6.7. Diagrama de flujo de la columna de destilación.</i>	115
<i>Figura 7.1. Principales componentes del DSS.</i>	128
<i>Figura 7.2. Definición de un equipo tipo columna.</i>	129
<i>Figura 7.3. Modelo matemático generado automáticamente por el MGM.</i>	131

## INDICE DE TABLAS

<i>Tabla 4.1. Variables del modelo (Joris &amp; Kalitventzeff, 1986).</i>	68
<i>Tabla 4.2. Conjuntos de asignación encontrados por GS-FLCN para el caso de estudio I.</i>	69
<i>Tabla 4.3. Lista de las unidades del proceso.</i>	70
<i>Tabla 4.4. Resultados para la planta de producción de etano.</i>	71
<i>Tabla 4.5. Tiempos de ejecución de GS-pFLCN.</i>	79
<i>Tabla 4.6. Speed-Up y Eficiencia logrados con GS-pFLCN.</i>	80
<i>Tabla 4.7. Corridas de GS-pFLCN para la planta de síntesis de amoníaco con diferentes BF.</i>	81
<i>Tabla 5.1. Tiempos de ejecución de CDHG y GS-FLCN para la planta de etano.</i>	94
<i>Tabla 5.2. Conjuntos de asignación hallados por el Enfoque Híbrido y por GS-FLCN.</i>	97
<i>Tabla 6.1. Resultados del análisis de observabilidad para el caso de estudio I.</i>	115
<i>Tabla 6.2. Resultados del análisis de observabilidad para el caso de estudio II.</i>	118
<i>Tabla 6.3. Resultados del análisis de observabilidad para el caso de estudio III.</i>	120

## INDICE TEMÁTICO

### A

Algoritmo Modificado	60, 62, 66
algoritmos de clasificación de variables	43
técnicas estructurales	49
técnicas no estructurales	49
análisis de instrumentación	41
análisis de observabilidad	14, 44
enfoque orientado a ecuaciones	46
enfoque orientado a topología	46
árbol de expansión	22
arista	19
arista de avance	25
arista de cruce	25
arista de retroceso	25
arista saliente	25

### B

BF	72, 80, 84, 98, 143
bigrafo	<i>Véase grafo bipartito</i>
bloques no reducibles	38
búsqueda en profundidad	23

### C

caminata ( <i>walk</i> )	21
caminata alternante	31
caminata cerrada ( <i>closed walk</i> )	21
caminata dirigida	22
camino ( <i>path</i> )	21
camino dirigido	22
cardinalidad	32
CD	<i>Véase conjuntos de desacople</i>
CDHG	84, 85, 86, 87, 91, 94, 95, 96, 97, 98, 99, 100, 121, 144, 145, 177
ciclo ( <i>cycle</i> )	21
ciclo dirigido	22
ciclo en un hipergrafo	39
circuito ( <i>circuit</i> )	21
circuito dirigido	22
columna base	62, 67, 68, 73

componentes conexas	22
componentes fuertes	22
conjuntos de desacople	135, 138, 147
construcción de la submatriz de ocurrencia	59
corrientes	46
<i>Cycle Detection on Hypergraphs</i>	<i>Véase CDHG</i>

## D

DCB	53, 54, 56, 57, 66, 67, 68, 69, 72, 73, 81, 142, 143
decision support system	<i>Véase DSS</i>
Depth-First Search	<i>Véase búsqueda en profundidad</i>
descomposición de Dulmage y Mendelsohn	37
descomposición fina	108
Descomposición Gruesa	106
Detección de Ciclos en Hipergrafos	<i>Véase CDHG</i>
detección de redundancias	44
DFS	<i>Véase búsqueda en profundidad</i>
DFSCCD	28
diagrama de flujo	46
digrafo	<i>Véase grafo dirigido</i>
DSS	125, 126, 127, 128, 129, 132, 134, 135, 136, 138, 139, 148, 149, 178

## E

ecuaciones	
ecuaciones asignadas	44
ecuaciones no asignadas	44
ecuaciones redundantes	44
eficacia	45
eficiencia	45
EH	96, 97, 98, 99
eliminación de Gauss-Jordan	51
enfoque híbrido	<i>Véase EH</i>
equipos de procesos	46
estrategia global con primer nodo menos conectado	<i>Véase GS-FLCN</i>
expansión forestal en profundidad	23
expansión forestal saliente en profundidad	24

## F

factores de ramificación	<i>Véase BF</i>
FLCN	60, 62, 63, 64, 67, 77, 94, 96, 117
flowsheet	<i>Véase diagrama de flujo</i>
FTB	<i>Véase forma triangular en bloques</i>
función EncuentraAutoAristas	86

**G**

GENOBS	49
GENRED	49
global strategy with first least-connected node	<i>Véase GS-FLCN</i>
grafo	19
grafo bipartito	30
grafo conectado	22
grafo débilmente conectado	23
grafo dirigido	20
grafo fuertemente conectado	22
grafo no dirigido	19
grafo subyacente	21
hipergrafo	38
GS-FLCN	56, 57, 64, 66, 68, 69, 72, 73, 74, 76, 77, 80, 81, 82, 84, 91, 94, 95, 96, 97, 98, 99, 100, 109, 114, 115, 116, 117, 118, 119, 120, 121, 122, 132, 133, 137, 142, 143, 144, 145, 146
GS-pFLCN	76, 77, 78, 79, 82, 85, 100, 143

**H**

hipergrafo	
hipergrafo cuadrado	40
hipergrafo parcial	39
hipergrafo	
hipergrafo inducido cerrado	40

**L**

LAN	<i>Véase red de área local</i>
-----	--------------------------------

**M**

matriz	
forma triangular en bloques	34
matriz bireducible	34
matriz de adyacencia de un digrafo	33
matriz de adyacencia de un grafo	33
matriz de incidencia de un hipergrafo	39
matriz reducible	34
método de bloque de columna dominante	<i>Véase DCB</i>
método directo	101, 102, 103, 104, 114, 119, 120, 132, 145, 146
método directo para análisis de observabilidad	101, 114, 116, 117, 118, 120, 121, 122, 146
método directo para análisis de observabilidad	115
modelo de estado estacionario	43
ModGen	71, 128, 180

**N**

nodo 19

**O**

orden de un grafo 19

orden de un hipergrafo 39

**P**

Parallel Virtual Machine (PVM) 77

pareamiento 31

    pareamiento maximal 31

PMB 35, 36, 106

porcentaje de variables asignadas *Véase pav*

procesamiento distribuido 76

procesamiento paralelo 76

proceso Haber-Bosch 78

pva 71, 72, 73, 95, 98, 99, 115, 117

**R**

raíz de la componente fuerte 27

rango de aplicación 45

rastro dirigido 22

red de área local 76

reglas heurísticas 88, 91, 92, 93, 95, 97, 99, 121, 144

revamp 42

robustez 54

**S**

sendero *Véase camino*

sendero alternante 31

sendero aumentado 32

simulated annealing 93, 144

sistema de soporte de decisión *Véase DSS*

subconjunto de asignación 52

subgrafo 20

    subgrafo de expansión 22

    subgrafo inducido 20

subgrafo inducido de un hipergrafo 39

Subrutina 2 60, 61, 66, 68

subrutina BuscarHGP 87

subrutina ChequearNxN 90

Subrutina N 62, 66

subrutina TestCiclo 89

<b>T</b>
----------

tamaño de un grafo	19
TCCD	28, 36, 108, 109
test de admisión	60, 61, 66, 71, 74, 103, 109, 110, 111, 112, 113, 121, 145
threshold	<i>Véase umbral</i>
topología de un proceso	46
transversal llena	34
triangularización hacia adelante	58

<b>U</b>
----------

umbral	77
--------	----

<b>V</b>
----------

variables	
variables no observables	43
variables no redundantes	43
variables observables	43
variables redundantes	43