XIV Conferencia Latinoamericana de Informática
17avas. Jornadas Argentinas de Informática
e Investigación Operativa
Buenos Aires, Setiembre 1988.

IMPLEMENTACION DE UN MODULO OPTIMIZADOR EN UN SIMULADOR DE PROCESOS QUIMICOS

Dr. Jorge M. Montagna
Dr. Oscar A. Iribarren

INGAR - Instituto de Desarrollo y Diseño
Avellaneda 3657 - (3000) Santa Fe
TE (042) 34451 - 45136

RESUMEN

Hasta hace pocos años, la optimización de un proceso químico utilizando un simulador de procesos modular secuencial era una tarea que demandaba un excesivo tiempo de computación, por lo que prácticamente no se usaba. Con la aplicación del método de Programación Cuadrática Sucesiva de Powel se logra reducir sustancialmente el tiempo de CPU empleado. Este método se ha implementado en el simulador SIMBAD, lo cual ha requerido el análisis de una serie de elementos. Se hizo hincapié en el manejo de la información mediante un Administrador de Base de Datos, pues el módulo optimizador interactúa con el simulador a través de la base de datos. Esto posibilita la aplicación de distintas estrategias para la optimización. Además, la estructura del SIMBAD, que maneja todos los elementos involucrados como programas ejecutables, simplifica la resolución de la optimización. Se ha generado un esqueleto que posibilitará el análisis e incorporación de los distintos elementos de este método: cálculo de gradientes, optimización univariable, resolución del problema cuadrático, etc. Dada la modularidad y el enfoque centrado en los datos con que fue desarrollado el SIMBAD, no ha sido necesario alterar sus elementos para realizar esta información.

1. Introducción

La simulación de procesos por computadora es una herramienta muy utilizada en el desarrollo de un proceso químico. Existen para ello un gran número de simuladores con avanzado desarrollo que calculan los balances de materia y energía de la planta. Casi todos los de uso industrial utilizan para la resolución el enfoque modular secuencial en el cual se resuelven las ecuaciones que modelan el funcionamiento de cada equipo en una subrutina, teniendo como dato las corrientes de entrada y los parámetros del mismo, y calculando sus corrientes de salida. Según aparezcan los equipos que componen la planta en la secuencia de resolución, son llamadas las distintas subrutinas. Las ventajas de este enfoque son, fundamentalmente, su robustez y confiabilidad, a partir del avance en los métodos y algoritmos para resolver cada equipo, y la posibilidad de seguir fácilmente el desarrollo de la simulación al coincidir el modelo computacional con la estructura de la planta.

Uno de los mayores inconvenientes del enfoque modular secuencial está en que el problema de la optimización demanda tiempos de computación tan elevados que imposibilitan su resolución. Normalmente la planta se maneja como si fuera una "caja negra". El módulo optimizador no tiene ninguna relación directa con la ejecución de la simulación y por lo tanto la evaluación de cada uno de los puntos de la sucesión del algoritmo requiere una simulación completa de la planta para poder calcular la función objetivo y las restricciones. Se utilizan métodos de busqueda directa.

Se da un cambio significativo con los trabajos de Biegler y Hughes (1981, 1982, 1985). Por un lado estos autores utilizan el método de Programación Cuadrática Sucesiva (PCS) de Powell (1977), pero además se abandona el esquema en el cual el simulador es independiente del optimizador y por el contrario hay ahora una relación entre ambos. Una serie de trabajos continúan hasta el presente en esta línea, buscando hacer cada vez más confiable, eficiente y rápida la resolución de la optimización de una planta química (Kisala (1985), Biegler (1985), Lang y Biegler (1986), etc).

En este trabajo se describe la implementación de un módulo optimizador para el simulador de procesos SIMBAD (Montagna y otros

(1987), Vecchietti y otros (1987) y Leone y otros (1987)). Tomando como base los desarrollos mencionados se ha utilizado el método PCS. Teniendo en cuenta que el SIMBAD utiliza para el manejo de la información un Administrador de Base de Datos (DBMS) se han aprovechado también en este caso las ventajas derivadas de ello, sobre todo en el sentido de mantener un enfoque centrado en los datos en su desarrollo. Además la estructura modular y flexible del SIMBAD, permite la fácil implementación y evaluación de distintas alternativas para los diferentes pasos de este algoritmo: cálculo de gradientes, optimización univariable, etc. y de las diferentes estrategias de resolución.

2. Optimización de procesos químicos

En principio, la simulación de un proceso químico es la resolución de un gran sistema de ecuaciones que modela la planta. Trabajando con el enfoque modular secuencial se reduce a una serie de subproblemas ordenados de acuerdo a la secuencia de resolcuión. Dado que cada subrutina que calcula un equipo requiere como dato las corrientes de entrada y como en las plantas de procesos químicos son comunes los reciclos, es necesario recurrir en casi todos los casos a métodos iterativos para la resolución. Estos parten de rasgar un conjunto de corrientes, proponer valores para las mismas e iterar repetidamente hasta alcanzar la convergencia. La figura 1 muestra un ejemplo muy simple de un trabajo de Biegler y Hughes (1982), en el cual se ha elegido como corriente de corte la 2, pues no existe ningún equipo que pueda ser resuelto por conocerse todas sus entradas. Proponiendo valores para esta corriente, y recogiendo los valores calculados para la misma del equipo IV, se converge la planta luego de algunas iteraciones.

La simulación puede resumirse entonces como:

$$h(y) = 0 \qquad (1)$$

donde h son las ecuaciones que exigen la convergencia de las corrientes de corte, e y las variables de estas últimas. El resto de las variables se obtiene como resultado intermedio del cálculo de los módulos.

Hasta el presente el problema de optimización se planteaba como:

PROCESO FLASH SIMPLE

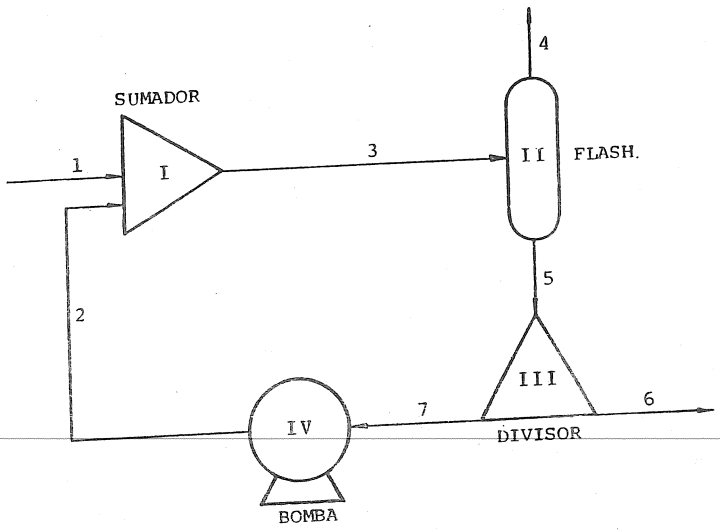

FIGURA 1

$$\text{Min } F(x)$$

sujeto a: (2)

$$c(x) = 0$$
$$g(x) \geq 0$$

donde c y g son las restricciones impuestas a las variables de diseño $\bar{x}$. Este planteo corresponde a la utilización del simulador como una caja negra, pues para cada x se efectúa una simulación completa de la planta, esto es, se resuelve (1).

En el nuevo enfoque se reunen la simulación y la optimización pues se trabaja con el siguiente problema:

$$\text{Min } F(x,y)$$

sujeto a:

$$h(x,y) = 0$$
$$c(x,y) = 0 \qquad (3)$$
$$g(x,y) \geq 0$$

o sea que se optimiza la función objetivo F sujeta a las restricciones h, c y g, las cuales se cumplen en el óptimo, en particular h que exige la convergencia de las corrientes de corte o sea la resolución de los balances de materia y energía del proceso.

De acuerdo a la forma en que se resuelve (3) surgen distintas estrategias: camino no factible (Biegler y Hughes (1982)), camino factible (Biegler y Hughes (1985)) e híbridos entre ambos (Kisala (1985)). Se diferencian en las iteraciones que para un dado x se realizan con distintos valores de y con (1). En el caso de los métodos de camino factible en cada iteración de la optimización se resuelve la simulación. Para camino no factible, se convergen al mismo tiempo ambos problemas. La ventaja del camino factible es que, si la optimización fracasa, se cuenta con una solución de la planta que, si bien no es óptima, permite realizar distintos análisis. Sin embargo, si la optimización concluye, se ha gastado tiempo de CPU resolviendo la simulación en puntos alejados del óptimo. Conclusiones opuestas se pueden obtener para camino no factible. Los métodos híbridos, si bien no exigen la convergencia de la simulación como en camino factible, permiten la aproximación de la solución realizando un determinado número

de iteraciones sobre la secuencia de resolución de la planta.

3. Método de Programación Cuadrática Sucesiva

Teniendo en cuenta el problema (3) el método de Programación Cuadrática Sucesiva (PCS) determina en cada iteración i una dirección de búsqueda d a partir del problema cuadrático:

$$\text{Min } \nabla F(x_i,y_i)^T d + \frac{1}{2} d^T B d$$

sujeto a:

$$\begin{aligned} h(x_i,y_i) + \nabla h(x_i,y_i)^T d &= 0 \\ c(x_i,y_i) + \nabla c(x_i,y_i)^T d &= 0 \\ g(x_i,y_i) + \nabla g(x_i,y_i)^T d &\geq 0 \end{aligned} \qquad (4)$$

La matriz hessiana B se obtiene aplicando la fórmula BFGS (Dennis y Moré (1977)). Sobre la dirección d obtenida de (4) se realiza una optimización univariable para determinar la longitud del paso en esta dirección. Se utiliza una función de penalidad que incluye un término de la función objetivo del problema original más una combinación no lineal de las restricciones violadas, siendo los factores de cada una de ellas función de los multiplicadores de Kuhn-Tucker de las dos últimas iteraciones.

4. Estructura de datos

SIMBAD utiliza una base de datos jerárquica para almacenar la información relativa a la simulación. Pese a ser un sistema network, dado el tipo de relaciones que se precisa manejar, con una estructura jerárquica ha sido posible diseñar el esquema de la base de datos (Vecchietti y otros (1987)). Básicamente éste tiene al tope información general de la planta (nombre, número de equipos, compuestos y corrientes, tipo de resolución, etc) y, en un segundo nivel colgada de la misma grupos de datos de corrientes, equipos y compuestos.

La implementación de la optimización requirió una mínima

modificación del esquema anterior. En la información general se agregaron items relativos a la optimización: tope y contador de iteraciones para la optimización global y la univariable, error máximo permitido, factor de perturbación, etc, y en el segundo nivel, información de las variables de optimización (figura 2).

El esquema logrado no ha implicado ningún cambio en ningún programa al variar la estructura de datos, pues las modificaciones son, desde el punto de vista resolutivo, independientes de lo existente. El único gasto ha sido el compilado y encadenado de todos los programas del simulador para generar la nueva área de trabajo de la base de datos.

Más allá de detallar las ventajas generales de la base de datos o del diseño efectuado en este caso, conviene puntualizar que se realizó un desarrollo centrado en los datos, tanto para el simulador SIMBAD original, como para el optimizador. En el primer caso permitió que la incorporación del módulo de optimización no implicara ningún cambio en el resto de los programas a partir de la flexibilidad con que fue diseñado. En el segundo caso, posibilitará la incorporación y evaluación de distintos elementos para cada uno de los pasos del método PCS y distintas estrategias de optimización.

5. Ejecución de la optimización

SIMBAD utiliza para la simulación dos archivos de comandos anidados. El primero, denominado Archivo Principal de Ejecución (APE) se encarga, principalmente, de generar el Archivo de Operación (AO) teniendo en cuenta la información almacenada en la base de datos en la etapa de carga y durante el preprocesamiento. El AO incluye los comando de ejecución de los programas de los equipos de la planta y otros auxiliares.

La figura 3 es el AO para la simulación de la planta de la figura 1 denominada FLASH1. Al comienzo se abre el fichero FLASH1.TRB que en ciertos casos incluye una cadena POOj de modo de ir a la ejecución del equipo ubicado en la posición j en la secuencia de resolución. Este dispositivo permite saltear la corrida de ciertos equipos, evitando un gasto innecesario en aquellos casos en que se ha realizado una corrida parcial y han surgido inconvenientes que obligaron a detener la

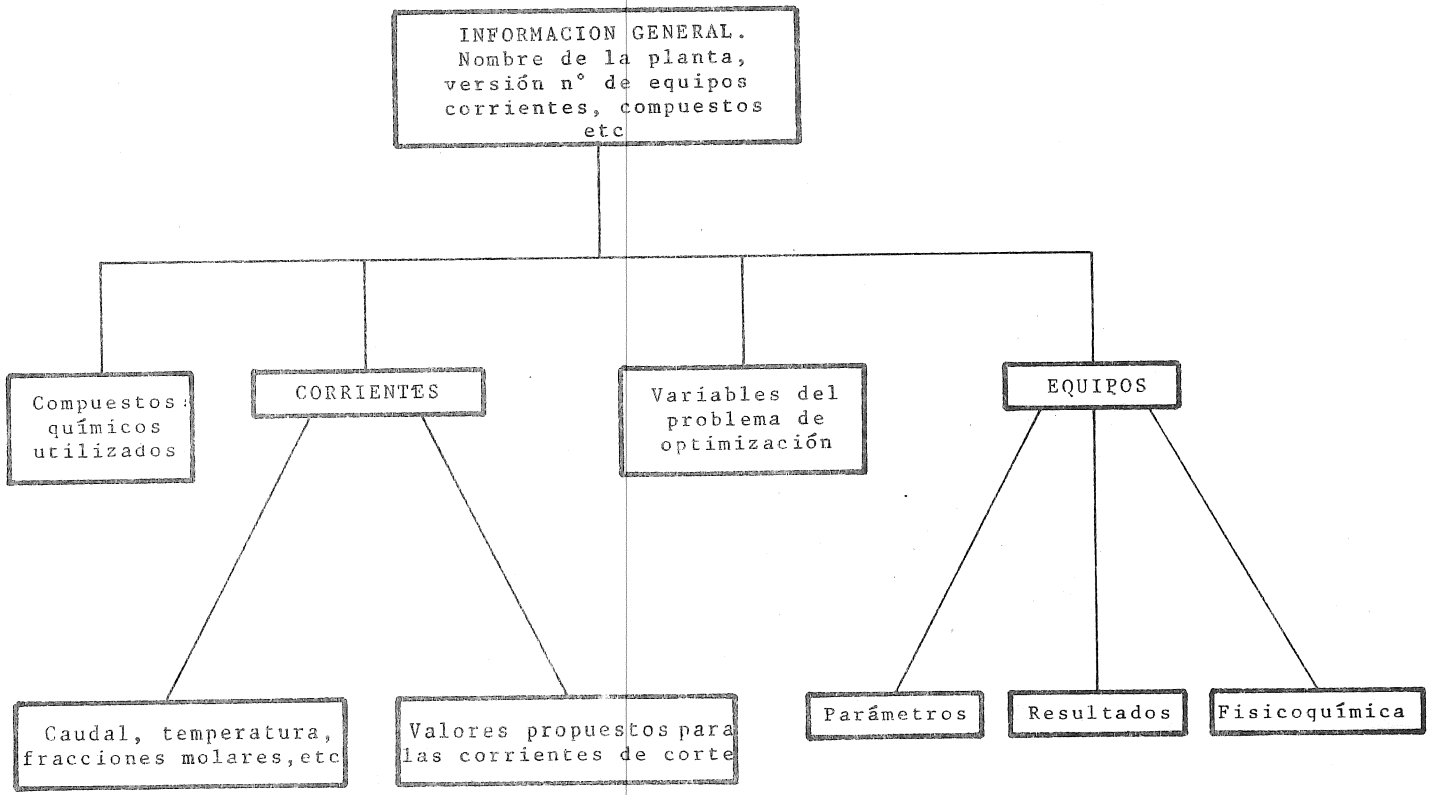

FIGURA 2.- Esquema General de la Base de Datos.

```
$ SET NOVERIFY
$ ON CONTROL_Y THEN GOTO JOUR
$ OPEN/READ/ERROR= P001 TRABAJO FLASH1.TRB
$             READ TRABAJO VAYA
$             CLOSE TRABAJO
$             DELE FLASH1.TRB;*
$             GOTO 'VAYA'
$ CORRE100:
$             CONTINUE
$ P001:
$         R [GSIMBAD.FINAL]NS1
$ P002:
$         R [GSIMBAD.FINAL]FV1N2
$ P003:
$         R [GSIMBAD.FINAL]ND1
$ P004:
$         R [GSIMBAD.FINAL]BV1N1
$ R [GSIMBAD.FINAL]CONVER
$ OPEN/READ ENTRADA FLASH1.CNV
$             READ ENTRADA KO
$             CLOSE ENTRADA
$ IF KO.EQS."A" THEN GOTO  CORRE100
$ DELETE FLASH1.CNV;*
$ R [GSIMBAD.FINAL]MASA
$ R [GSIMBAD.FINAL]ID
$ R [GSIMBAD.FINAL]IC
$ R [GSIMBAD.FINAL]IR
$ R [GSIMBAD.FINAL]DOC
$ EXIT
$ JOUR:
$ @SEGURO
$ EXIT
```

FIGURA 3.- Archivo de Operación para la simulación de la planta FLASH1 con el SIMBAD.

simulación. Luego, el AO tiene los comandos de ejecución de los distintos equipos ordenados según la secuencia de resolución. En este ejemplo se simulan un sumador, un flash, un divisor y una bomba con los programas NS1, FV1N2, ND1 y BV1N1, opciones seleccionadas en la carga de datos. Luego el programa CONVER, utilizando el método de sustitución directa o el de aceleración de Wegstein, efectúa la convergencia de la corriente de corte 2. Si no se ha alcanzado la convergencia, se retoma la ejecución de los equipos desde CORRE100 y, en caso contrario, se imprimen los datos y resultados con los programas ID, IC e IR.

Las ventajas de una estructura como la descripta son, principalmente, un eficiente uso de memoria central y de los recursos del sistema, y una estructura sumamente modular y flexible que permite la implementación de distintas estrategias de resolución sin modificar los elementos del simulador. Todos éstos interactúan a través de la base de datos, de la cual retiran la información para el cálculo, y donde depositan los resultados para que sean utilizados por las siguientes aplicaciones.

La introducción de la opción de optimización no ha implicado cambios en la estructura del SIMBAD, teniendo en cuenta las ventajas ya mencionadas. El único programa que se altera es el incluido en el APE, encargado de generar el AO. Este debe contemplar la participación del módulo optimizador y otros cambios menores. La figura 4 es el AO para la optimización de la planta FLASH1. En primer lugar se ha reemplazado el programa CONVER por el OPTIMIZA que se encarga de la aplicación del método PCS. Es preciso reiterar que, de acuerdo a (3), al mismo tiempo que optimiza, se convergen las corrientes de corte al incluir $h(x,y)=0$ entre las restricciones. Además se ha cambiado la posición del rótulo CORRE100 pues antes, para converger las corrientes de corte, siempre se calculaban todos los equipos desde el comienzo de la secuencia de resolución. Para optimizar muchas veces es necesario simular una parte de dicha secuencia a partir de un equipo intermedio, por lo que el dispositivo con el fichero de extensión. TRB es muy útil. Por ejemplo, si se desea calcular gradientes con respecto al coeficiente de partición del divisor III, luego de perturbar este valor en la base de datos, el programa OPTIMIZA carga en FLASH1.TRB la cadena P003, y sólo se corren para calcular dichos gradientes los equipos III y IV.

La figura 5 muestra esquemáticamente la operación del optimizador.

```
$ SET NOVERIFY
$ ON CONTROL_Y THEN GOTO JOUR
$ CORRE100:
$ OPEN/READ/ERROR= P001 TRABAJO FLASH1.TRB
$             READ TRABAJO VAYA
$             CLOSE TRABAJO
$             DELE FLASH1.TRB;*
$             GOTO 'VAYA'
$
$ P001:
$         R [GSIMBAD.FINAL]N61
$ P002:
$         R [GSIMBAD.FINAL]FV1N2
$ P003:
$         R [GSIMBAD.FINAL]ND1
$ P004:
$         R [GSIMBAD.FINAL]BV1N1
$ R [GSIMBAD.FINAL]OPTIMIZA
$ OPEN/READ ENTRADA FLASH1.OPT
$             READ ENTRADA KO
$             CLOSE ENTRADA
$ IF KO.EQS."A" THEN GOTO  CORRE100
$ DELETE FLASH1.OPT;*
$ R [GSIMBAD.FINAL]MASA
$ R [GSIMBAD.FINAL]ID
$ R [GSIMBAD.FINAL]IC
$ R [GSIMBAD.FINAL]IR
$ R [GSIMBAD.FINAL]DOC
$ EXIT
$ JOUR:
$ @SEGURO
$ EXIT
```

FIGURA 4.- Archivo de Operación para la optimización de la planta FLASH1 con el SIMBAD.

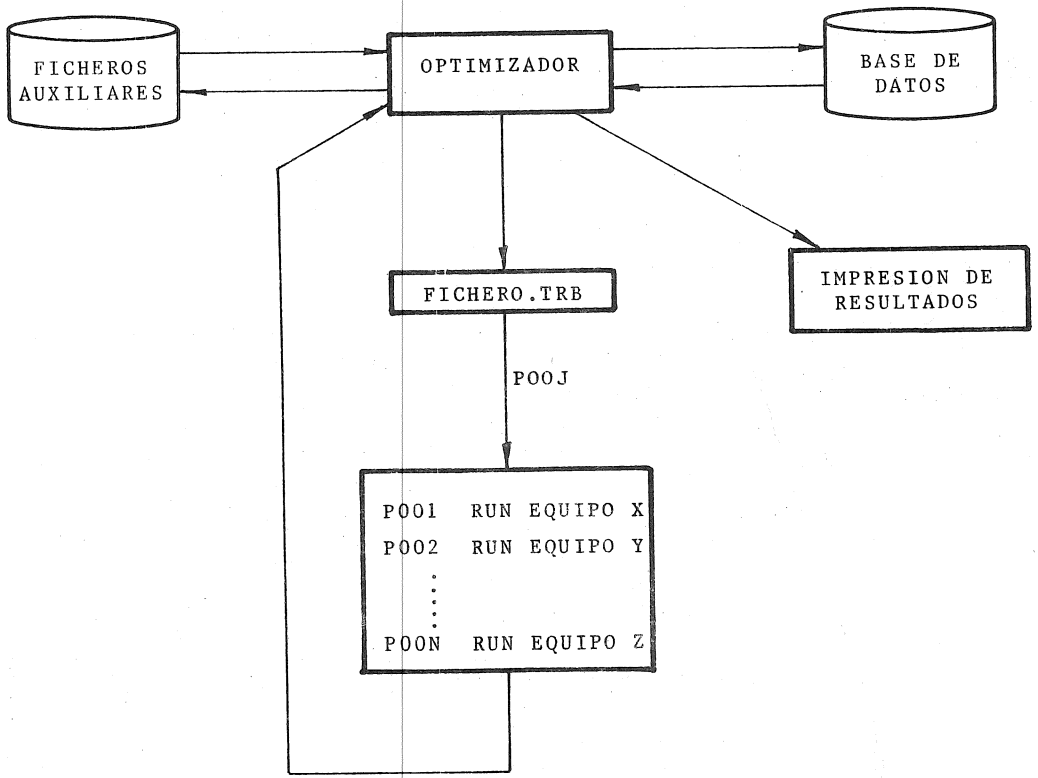

FIGURA 5.- Ejecución de la optimización.

Este interactúa con la base de datos, pero también cierta información la maneja a través de ficheros auxiliares por la volatilidad de esos valores. La relación con la secuencia de resolución se realiza en todos los casos (cálculo de gradientes, optimización univariable, etc.) a través del fichero de extensión .TRB.

La estructura implementada conserva la modularidad del SIMBAD. Además permite cambiar con facilidad distintas estrategias: camino factible, no factible o híbridos intermedios, incorporar por ejemplo módulos simplificados para el cálculo de los gradientes, etc.

6. Futuros desarrollos y conclusiones

El trabajo presentado describe la estructura básica del módulo optimizador del SIMBAD. Se trata de un esqueleto sobre el cual se irán incorporando distintas opciones en el futuro. Muchas de estas corresponden a variantes del método PCS y a su aplicación al caso concreto de procesos químicos. Existen un número de parámetros y factores que afectan la performance y eficiencia del método que es preciso determinar con la mayor exactitud posible, más si se tiene en cuenta que para este tipo de problemas una evaluación de la función objetivo y las restricciones en cierto punto demandan un considerable tiempo de CPU. Los elementos a analizar son:

- Cálculo de los gradientes: por perturbación directa (única opción disponible al momento), regla de la cadena, jacobiano analítico, módulos simplificados, etc.
- Optimización univariable: selección de la función de penalidad y los factores para cada restricción violada (al momento se usa lo planteado por Powell (1977)), criterio de terminación, etc.
- Inicialización del problema
- Escalado del problema
- Factor de perturbación
- Selección de las corrientes de corte

Además, cada uno de los puntos anteriores está incluido en una estrategia de resolución: camino factible, no factible e híbridos

intermedios. Todas las anteriores trabajan con el flujo de información para la resolución coincidiendo con el flujo de materia y energía en la planta. Si se altera para la resolución el flujo de información buscando aquel que favorezca la resolución, utilizando el planteo de Montagna e Iribarren (1988) se puede obtener una reducción en el tiempo de CPU requerido.

Se cuenta por lo tanto con la estructura del módulo optimizador que permitirá la evaluación de distintas alternativas de modo de obtener un eficiente algoritmo para la optimización de procesos químicos. Este es un tema en el cual se está trabajando activamente en distintos centros y que ha llevado a que ya hayan aparecido las primeras versiones de simuladores comerciales incluyendo versiones iniciales de optimización general de procesos.

REFERENCIAS

- L. Biegler, Improved infeasible path optimization for sequential modular simulators - I. The interface, Comp. and Chem. Engng., 9, 3, 245 (1985).

- L. Biegler y R. Hughes, Approximation programming of chemical processes with Q/LAP, Chem. Engng. Prog., 77, 4, 76 (1981).

- L. Biegler y R. Hughes, Feasible path optimization with sequential modular simulator, Comp. and Chem. Engng., 9, 4, 379 (1985).

- L. Biegler y R. Hughes, Infeasible path algorithms for sequential modular optimization, AIChE J., 28, 6, 994 (1982).

- J. Dennis y J. Morê, Quasi-Newton methods, motivation and theory, SIAM Rev., 19 (1), 46 (1977).

- T. Kesala, Successive quadratic programming in sequential modular process flowsheet simulation and optimization, Doctoral Thesis Digest, M.I.T., Cambridge, E.E.U.U. (1985).

- Y. Lang y L. Biegler, A Unified Algorithm for Flowsheet Optimization, EDRC-06-18-86, Engineering Design Research Center, Carnegie Mellon University, E.E.U.U. (1986).

- H. Leone, T. Melli, J. Montagna, A. Vecchietti y R. Cerro, SIMBAD: A steady state process simulator associated to a DBMS. III. The physico-chemical properties package, Comp. and Chem. Engng., 11, 3, 217 (1987).

- J. Montagna, H. Leone, T. Melli, A. Vecchietti y R. Cerro, SIMBAD: A steady state process simulator associated to a DBMS. I. The executive system, Comp. and Chem. Engng., 11, 63 (1987).

- J. Montagna y O. Iribarren, Optimal computation sequence in the simulation of chemical plants, Comp. and Chem. Engng., 12, 1, 71 (1988).

- M. Powell, A fast algorithm for nonlinearly constrained optimization calculations, trabajo presentado en la Dundee Conference on Numerical Analysis, Escocia (1977).

- A. Vecchietti, H. Leone, T. Melli, J. Montagna y R. Cerro, SIMBAD: A steady state process simulator associated to a DBMS. II. The structure of memory, Comp. and Chem. Engng., 11, 5, 519 (1987).