UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

EVALUACIÓN TÉCNICA DE CÓDIGOS COMPUTACIONALES PARA LA OPTIMIZACIÓN DE LA OPERACIÓN DE CORTO PLAZO EN EL SING

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

CRISTIAN LEONARDO ROMERO HERNÁNDEZ

PROFESOR GUÍA:
LUIS VARGAS DIAZ

MIEMBROS DE LA COMISIÓN:
RODRIGO PALMA BEHNKE
ÓSCAR MOYA ARAVENA

SANTIAGO DE CHILE
MARZO 2008

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELECTRICISTA
POR: CRISTIAN ROMERO H.
FECHA: 26/03/2008
PROF. GUÍA: SR. LUIS VARGAS D.

"EVALUACIÓN TÉCNICA DE CÓDIGOS COMPUTACIONALES PARA LA OPTIMIZACIÓN DE LA OPERACIÓN DE CORTO PLAZO EN EL SING"

El objetivo general del presente trabajo de título es realizar, mediante la aplicación de criterios técnicos de ingeniería, una evaluación técnica del desempeño de los algoritmos de Relajación Lagrangeana (RL) y Branch and Bound (B&B) en la búsqueda de soluciones para el problema de optimización de corto plazo en el sistema eléctrico interconectado del norte grande (SING).

En la primera parte de la memoria se muestra el planteamiento general del problema de optimización de la operación de corto plazo, el cual corresponde a un problema de optimización entero-mixto y un conjunto de restricciones lineales mediante las cuales se establecen las características técnicas del sistema. Por otra parte, la función objetivo de dicho problema de optimización corresponde a la minimización de los costos asociados a la operación de las unidades en el horizonte de tiempo evaluado.

Posteriormente, se muestra una revisión del estado del arte presentando algunas de las principales técnicas utilizadas para resolver este tipo de problema: Lista de Prioridad, Programación Dinámica, *Unit Decommitment*, RL, Método de Benders, B&B y Algoritmos Genéticos.

Para realizar la evaluación sobre los algoritmos de RL y B&B, se realizan programas en Matlab de dichos métodos con el objeto de realizar pruebas que permitan efectuar un análisis comparativo de los rendimientos de ambos algoritmos.

Se aplican dichos programas para resolver problemas de predespacho en un modelo reducido del SING. De esta forma se puede observar el rendimiento de cada algoritmo respecto de su capacidad de obtener soluciones factibles, calidad de las soluciones, uso de heurística para generar soluciones y tiempos de ejecución requeridos. Adicionalmente, se puede estudiar la flexibilidad de ambos algoritmos para considerar restricciones de mayor complejidad y sus limitaciones para resolver predespacho en sistemas de dimensiones reales.

Se concluye que el algoritmo que presenta un rendimiento que permite resolver de manera más eficiente el problema de predespacho en el SING corresponde al algoritmo RL, lo anterior debido principalmente a los tiempos de ejecución requeridos para su aplicación en sistemas de dimensiones reales y a que las soluciones generadas presentan una precisión del orden del 99% respecto a las soluciones generadas por el otro algoritmo. Adicionalmente, se puede acotar que las actuales políticas de operación aplicadas en el SING no representan una gran complejidad de programación y por lo tanto, la heurística requerida no presenta una complejidad adicional.

AGRADECIMIENTOS

En primer lugar deseo agradecer a mi familia y muy especialmente a mis padres, Enrique y Doris, quienes han sido un ejemplo por su esfuerzo y una guía moral para mi formación como profesional y persona, este logro se lo debo a ellos.

También quiero expresar mi gratitud a mi esposa Jéssica por acompañarme, apoyarme y motivarme en esta última etapa que resultó tan difícil.

A todos los profesores y funcionarios de la facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile, muchas gracias, por todo lo que me entregaron en este largo camino.

Finalmente, agradezco a los profesores miembros de mi comisión: Dr. Luis Vargas, Dr. Óscar Moya y Dr. Rodrigo Palma, por su apoyo y buena disposición.

ÍNDICE DE CONTENIDOS

1 INTRODUCCIÓN

La operación económica de un sistema eléctrico interconectado corresponde a un problema de optimización, mediante el cual se intenta encontrar el mínimo costo asociado a la producción de energía para satisfacer los requerimientos de los consumidores de energía eléctrica y las pérdidas de transmisión propias del sistema. Asimismo, para que la planificación de la operación de corto plazo sea óptima, en el sentido económico, se debe determinar la combinación óptima de unidades generadoras que deben operar en cada etapa del período de planificación y a su vez la producción de cada una de estas unidades, de manera de obtener el menor costo posible asociado a la generación de energía.

Sin perjuicio de lo anterior, la operación de un sistema eléctrico además de responder a criterios económicos, también debe considerar aspectos relativos a la seguridad y calidad de servicio, como lo es mantener el suministro total o en forma parcial (si consideramos operaciones de desprendimientos automáticos de carga) ante determinadas contingencias (fallas de unidades generadoras o componentes del sistema de transmisión).

Producto de lo anterior, la optimización de la operación de corto plazo de un sistema eléctrico corresponde a la minimización del costo de operación de las unidades generadoras, considerando restricciones de balance energético (demanda debe ser igual a la generación), restricciones técnicas de las unidades: límites máximos y mínimos de generación, tiempos mínimos y máximos de operación y detención, etc., y restricciones de seguridad.

Matemáticamente el problema de predespacho es un problema de programación entero mixta del tipo NP-completo, el cual presenta como variables continuas el nivel de producción de las unidades generadoras y las variables enteras corresponderán a las variables de decisión que determinan si una unidad es despachada o no.

Obtener la solución óptima de un problema de predespacho para un sistema de dimensiones reales es muy difícil, es por ello que se han desarrollado múltiples métodos de búsqueda de soluciones cercanas al óptimo para resolverlo. Entre ellos destaca el algoritmo de Relajación Lagrangeana como uno de los más exitosos.

El método de Relajación Lagrangeana corresponde a un algoritmo de descomposición, en el cual las restricciones del problema se clasifican en dos grupos: Restricciones de acoplamiento y restricciones individuales, las primeras corresponden a aquellas restricciones de igualdad o desigualdad que en una misma ecuación consideren variables asociadas a más de una unidad, por ejemplo la restricción de balance energético. Por el contrario, las restricciones individuales son aquellas restricciones de igualdad o desigualdad que pueden contener variables pertenecientes a distintos períodos de tiempo pero involucran variables asociadas a una sola unidad en cada ecuación, por ejemplo las restricciones de tiempos mínimos de operación y detención de las unidades.

Mediante esta clasificación de las restricciones, se crea la función objetivo del problema dual, la cual corresponde a la suma de la función objetivo del problema primal y las restricciones de acoplamiento ponderadas por factores no negativos denominados *multiplicadores de lagrange*. Asimismo considera como restricciones del problema dual las restricciones individuales y la no negatividad de los multiplicadores.

La estructura del problema dual definido permite resolver dicho problema mediante la descomposición en subproblemas de optimización, uno por cada unidad generadora. Estos subproblemas a su vez pueden ser resueltos, debido a que tienen una dimensión mucho menor, con algoritmos de optimización combinatorial de menor complejidad, comúnmente mediante *Programación Dinámica*.

Una importante propiedad dual sustenta este procedimiento, la cual indica que el óptimo de la función dual corresponderá siempre a una cota inferior del óptimo del problema primal, de esta forma, el algoritmo de Relajación Lagrangeana resuelve en forma iterativa los problemas dual y primal, actualizando las variables duales (multiplicadores) y primales en cada iteración, de manera de aproximar las soluciones de ambos problemas y obtener una solución cercana al óptimo. Es precisamente la diferencia entre ambas soluciones (dual y primal) la que determinará la convergencia del algoritmo, definiéndose la denominada *Brecha dual* como el porcentaje que representa la diferencia entre ambas soluciones respecto de la solución del primal.

La principal ventaja del método de *Relajación Lagrangeana* corresponde a su simplicidad matemática, dado que resuelve en cada iteración problemas de optimización de dimensiones mucho menores a las del problema total, esto tiene una consecuencia directa en los tiempos de ejecución requeridos para obtener soluciones con una aceptable aproximación al óptimo cuya brecha dual comúnmente presenta valores entre 1 a 2 %.

Por otra parte, otro tipo de algoritmos que también son muy conocidos y utilizados en la resolución del problema de predespacho corresponden a los métodos generales de programación entero mixta, principalmente los algoritmos denominados *Cortes de Benders* y *Branch and Bound*. Estos métodos consisten en resolver problemas relajados en los cuales se consideran las variables enteras como continuas con lo cual el espacio de soluciones del problema original corresponde a un subconjunto del espacio de soluciones del problema relajado. De esta forma y mediante la adición sucesiva de restricciones al problema original relajado se va generando una división del espacio de soluciones en subconjuntos disjuntos de búsqueda que deben conducir a encontrar soluciones factibles para el problema original. Estos métodos son capaces de obtener la solución óptima de problemas de predespacho, sin embargo, el principal inconveniente de este tipo de métodos es el esfuerzo computacional requerido para resolver problemas de gran escala.

El método de Branch and Bound consiste en generar un árbol de búsqueda cuya raíz corresponde al problema original, con las variables enteras relajadas (variables de decisión ON/OFF), a partir de este nodo raíz se generan subproblemas a los cuales se les agregarán progresivamente restricciones de igualdad para generar soluciones en espacios de búsqueda disjuntos que lleven a obtener una solución factible para el problema original de programación entera mixta.

Las combinaciones de cortes requeridos puede ser muy grande y la naturaleza enumerativa del método Branch and Bound incrementa enormemente el esfuerzo computacional.

El presente trabajo tiene como objetivo principal efectuar una evaluación técnica del desempeño de los algoritmos Relajación Lagrangeana y Branch and Bound para resolver el problema de predespacho, considerando en particular su aplicación a un modelo reducido del Sistema Interconectado del Norte Grande (SING).

Para conseguir este objetivo en primer lugar se presentará una descripción general del problema de predespacho y se realizará una revisión de los principales métodos utilizados para resolver este problema. Posteriormente se describirán en detalle los algoritmos de Relajación Lagrangeana y Branch and Bound aplicados a predespacho y se realizarán programas computacionales de ambos métodos para realizar aplicaciones experimentales en sistemas ficticios y finalmente en el modelo reducido del SING. Finalmente se concluirá acerca de los rendimientos observados.

2 ALCANCES DE LA MEMORIA

El objetivo principal de la presente memoria es realizar un análisis comparativo de las dos principales técnicas de optimización utilizadas para resolver el problema de predespacho de unidades generadoras de un sistema eléctrico interconectado: Relajación Lagrangeana y Branch and Bound.

Este análisis se basa en la comparación de programas desarrollados en Matlab para dicho propósito, estos programas aplican los algoritmos generales de optimización utilizando estas técnicas y teniendo como motor de optimización una función predefinida en Matlab.

Utilizando estos programas se realizarán pruebas que permitan un análisis del rendimiento de ambos algoritmos para resolver problemas de predespacho. Estas pruebas se efectuarán sobre sistemas ficticios de pocas barras (10 a 15) y finalmente sobre un sistema reducido del SING de 14 barras.

Si bien el análisis simplificado que se ha descrito en forma general permite obtener una idea general sobre las ventajas y desventajas que presentan ambos algoritmos, para tener una conclusión acabada de las características, rendimiento, limitaciones y capacidades de ambos algoritmos se debe realizar un estudio de "*Benchmarking*" utilizando software comerciales que utilicen estas técnicas. Esto último requiere de disponer de recursos importantes, desde los software que serán puestos a prueba hasta computadores con diferentes características para comparar el rendimiento en cada uno de ellos.

La presente memoria propone un análisis simple y con recursos limitados, por lo cual las conclusiones que se obtiene tienen un alcance acotado, pero que entrega una visión general.

3 OPTIMIZACIÓN DE LA OPERACIÓN DE CORTO PLAZO

3.1 Descripción del Problema

El problema que se desea resolver corresponde a la determinación la combinación óptima de unidades necesarias para satisfacer los requerimientos mínimos para la operación de un sistema eléctrico en un horizonte de tiempo determinado, usualmente denominado pre-despacho o Unit Commitment. El horizonte de tiempo común suele definirse entre 24 y 168 horas, normalmente dividido en intervalos horarios.

La resolución de este problema requiere que en cada una de las etapas del período de evaluación deben decidirse las unidades y los niveles de producción que garanticen la operación a mínimo costo para satisfacer los requerimientos de demanda en forma adecuada. Esto hace que el problema matemático resultante corresponda a una combinatoria con variables continuas y variables enteras, estas últimas asociadas a la decisión de tener en servicio o no una determinada unidad generadora, mientras que las variables continuas representarían aspectos tales como la potencia requerida en las centrales en un determinado instante de tiempo.

De esta forma, el problema antes descrito se puede presentar como una función objetivo, que representará el ámbito que se desea optimizar, por ejemplo los costos asociados a la operación, y un conjunto de restricciones que representarán las condiciones que se desean satisfacer.

A continuación se presenta el planteamiento general del problema de Unit Commitment, función objetivo y restricciones básicas.

3.2 Planteamiento General del Problema

3.2.1 Función Objetivo

La función objetivo del problema de optimización para un período de evaluación determinado será: Minimizar: Costos Operativos + Costos de Partida + Costos de Parada

La siguiente expresión representa la función objetivo del modelo matemático de programación entero-mixto básico para resolver el problema de pre-despacho para un período T:

$$Min\ F = \sum_{t}^{T}\sum_{n}^{N}\left(U_{n,t}\cdot FC(P_{n,t}) + S_{n,t}^{start} + S_{n,t}^{stop}\right) \qquad (1)$$

donde:

$U_{n,t}$: Variable binaria asociada al estado ON/OFF de la unidad *n* en el tiempo *t*.

$P_{n,t}$: Potencia asociada a la unidad *n* en el tiempo *t*.

$FC(P)$: Función de costos.

$S_{n,t}^{start}$: Costo de partida de la unidad *n* en el tiempo *t*

$S_{n,t}^{stop}$: Costo de detención de la unidad *n* en el tiempo *t.*

N: Número de unidades

T: Período de evaluación.

3.2.2 Restricciones del Planteamiento General

Se define como restricción de acoplamiento aquella que involucre en la misma ecuación, bien sea de igual o de desigualdad, variables de estado pertenecientes a más de una unidad generadora.

Se define como restricción individual aquella que puede involucrar variables pertenecientes a distintos períodos de tiempo pero correspondientes a una misma unidad generadora.

El problema de Unit Commitment contiene restricciones de acoplamiento e independientes, a continuación se detallan las principales separadas en estas dos categorías.

3.2.2.1 Restricciones de acoplamiento

1) Balance Energético:

La generación en cada instante de tiempo debe ser igual a la demanda del sistema más las pérdidas del sistema de transmisión:

$$\sum_{n=1}^{N} U_{n,t} \cdot P_{n,t} = D_t \qquad \forall\, t = 1,\ldots,T \qquad (2)$$

donde:

D_t : Demanda bruta del sistema en el tiempo *t.*

2) Capacidad de Reserva en Giro:

Para satisfacer los requerimientos de regulación de frecuencia ante variaciones de la demanda y perturbaciones en el sistema se requiere mantener en el sistema un monto determinado de reserva en giro.

$$\sum_{n}^{N} U_{n,t} \cdot \left(P_{n,t}^{\max} - P_{n,t}\right) \geq R^t \qquad \forall\, t = 1,\ldots,T \qquad (3)$$

3) Reserva en Giro del Sistema:

Se requiere que la suma de la contribución máxima de reserva que puede aportar cada unidad ($Rmáx_n$) que se encuentra en servicio sea suficiente para satisfacer el requerimiento de reserva del sistema.

$$\sum_{n}^{N} U_{n,t} \cdot Rmáx_n \geq R^t \qquad (4)$$

3.2.2.2 Restricciones Individuales

4) Capacidad de las Unidades:

Las unidades pueden generar en un rango definido por la potencia mínima o mínimo técnico y su potencia máxima.

$$P_{n,t}^{\min} \le P_{n,t} \le P_{n,t}^{\max} \qquad \forall\, n = 1,\ldots, N \quad \forall\, t = 1, \ldots, T \quad (5)$$

5) Costos de Partida:

Los costos de partida de las unidades generadoras se definen de la siguiente manera:

$$S_{n,t}^{start} = U_{n,t} \cdot \left(1 - U_{n,t-1}\right) \cdot \hat{S}_{n,t} \qquad (6)$$

donde $\hat{S}_{n,t}$ corresponde al costo de partida en frío, tibio o caliente de la unidad *n*, dependiendo del tiempo en que la unidad se ha mantenido fuera de servicio.

6) Costo de Detención:

Los costos de detención de las unidades generadoras se definen de la siguiente manera:

$$S_{i,t}^{stop} = U_{i,t-1} \cdot (1 - U_{i,t}) \cdot \breve{S}_{i,t} \qquad (7)$$

donde $\breve{S}_{n,t}$ corresponde al costo asociado a la detención de la unidad *i* cuando esta se encuentra operando.

7) Límite en Tasa de Toma y Descenso de Carga:

Las unidades generadoras pueden tomar carga a una tasa limitada, este limite corresponde al gradiente máximo de subida de la unidad y se representa en MW/min.

$$P_{n,t} - P_{n,t-1} \le \hat{G}_n^{UP} \cdot 60 \qquad \forall\, n = 1, \ldots, N \quad \forall\, t = 1, \ldots, T \qquad (8)$$

Asimismo, las unidades generadoras tienen un límite en la velocidad de descenso de su generación también representada en MW/min.

$$P_{n,t-1} - P_{n,t} \le \breve{G}_n^{DOWN} \cdot 60 \qquad \forall\, n = 1, \ldots, N \quad \forall\, t = 1, \ldots, T \ (9)$$

8) Limites de Tiempo:

- Tiempo mínimo en servicio: tiempo mínimo que la unidad debe permanecer en servicio cuando esta es sincronizada a la red ($\hat{t}_n^{on}$).
- Tiempo mínimo de apagado: tiempo mínimo que la unidad debe estar detenida inmediatamente después de salir de servicio ($\hat{t}_n^{off}$).

La restricción asociada se puede representar de la siguiente forma:

$$U_{n,t} \quad \begin{cases} 1 & si \quad t_{n,t}^{on} \leq \hat{t}_n^{on} \\ 0 & si \quad t_{n,t}^{off} \leq \hat{t}_n^{off} \end{cases} \quad \forall\, n = 1, \ldots, N \quad \forall\, t = 1, \ldots, T \qquad (10)$$

3.3 Principales Métodos de Optimización Empleados

3.3.1 Lista de Prioridad

La lista de prioridad corresponde al más sencillo de los métodos desarrollados para resolver el problema de Unit Commitment. Este método tiene una fuerte dependencia de la heurística y se basa en utilizar una lista de prioridad para seleccionar las unidades que deben operar en cada intervalo de tiempo.

Las utilizadas para generar la lista de mérito se basan en:

1. Basadas en costos de producción a plena carga.
2. Basadas en costos incrementales
3. Basadas en costos incrementales con costos de partida
4. Listas de prioridad dinámicas.

Sin duda la más interesante y de mayor complejidad corresponde a las listas de prioridad dinámicas, las cuales mediante la utilización de heurísticas realizan ajustes sucesivos sobre el conjunto de unidades seleccionadas mediante una lista de mérito típica, de manera de satisfacer de manera simultánea para cada etapa todas las restricciones del problema, en particular las relativas a tiempos mínimos de operación y detención y el balance energético.

Una de las característica de este método consiste en determinar un conjunto de unidades que se encuentran en la clasificación "*must run*", característica dada por su gran tamaño y por su bajo costo de operación. Dichas unidades se consideran en un estado fijo (en servicio) y por lo tanto, no participan en el algoritmo de búsqueda de la solución, acotando de esta forma el espacio de búsqueda.

Las principales ventajas de este método son la simplicidad y rapidez con que se obtienen soluciones y las desventajas más relevantes corresponden a su fuerte dependencia de la heurística y que entrega resultados alejados del óptimo real.

3.3.2 Programación Dinámica

A continuación se presenta una descripción general del algoritmo de Programación Dinámica respecto a su aplicación para resolver el problema de Unit Commitment.

El algoritmo de Programación Dinámica evalúa en forma sistemática un gran número de posibles decisiones con el fin de minimizar el costo total de un problema de predespacho de múltiples etapas.

Para realizar lo anterior, en cada etapa se evalúan todas las posibles combinaciones de unidades ($2^N - 1$ para un sistema de N unidades) que satisfacen tanto la demanda (que incluye la reserva en giro y las pérdidas de transmisión), como las restricciones técnica de las unidades (límites operacionales, tiempos mínimos de operación y detención, etc.), de esta forma se obtiene la combinación que minimiza el costo total en el horizonte de tiempo evaluado.

Dada la naturaleza enumerativa del método, la Programación Dinámica requiere de un gran tiempo de procesamiento, el cual crece exponencialmente con el tamaño del problema (número de unidades) y rápidamente alcanza niveles que la hacen inviable computacionalmente.

Producto de lo anterior, en la práctica para poder aplicar este método a la resolución del Unit Commitment en sistemas grandes, se han introducido muchas estrategias que involucran la heurística y utilizan listas de prioridad [1] (listas de mérito), con el objetivo de reducir los tiempos de procesamiento asociados.

El algoritmo conocido como DP – SC, utiliza una lista de prioridad estricta para la secuencia de búsqueda, de esta forma se reducen el número de combinaciones posibles en cada etapa. Otro método emplea una ventana de búsqueda fija, mediante la cual trunca (reduce) la lista de prioridad y evalúa solamente las combinaciones posibles dentro de dicha ventana. Este método es conocido como DP – TC y presenta un mejor desempeño en la búsqueda del óptimo que el DP – SC, sin embargo requiere mucho más tiempo de procesamiento.

Una alternativa presentada en [2] utiliza el algoritmo DP – TC, en donde se selecciona una ventana búsqueda de tamaño fijo *Wn* en la lista de prioridades, dicha ventana corresponde a un subconjunto de las unidades que participan del predespacho, tal como se muestra en la figura 1. Las unidades que se encuentran bajo el límite inferior de la ventana corresponderán a unidades *"must run"*, es decir unidades que debido a su tamaño y costo de producción deberían permanecer en servicio para cualquier nivel de demanda que presente el sistema en el horizonte de optimización. Por otra parte, el conjunto de unidades que se encuentran sobre el límite superior de la ventana, corresponderán a unidades que debieran permanecer fuera de servicio por que se encuentran indisponibles o unidades que debieran ser consideradas durante los *peak* de carga y estados de emergencia, dada su poca eficiencia.

Si el tamaño de la ventana es seleccionado de acuerdo con estos criterios, la solución óptima podría estar garantizada por la evaluación completa de todos los subconjuntos de todas las unidades generadoras disponibles con sus estados ON / OFF.

Desafortunadamente, dicho tamaño de ventana para un problema práctico es aun demasiado grande y la reducción del esfuerzo computacional podría no ser suficiente. Por lo tanto, una practica común es ejecutar el programa para una ventana de tamaño moderado, aún corriendo el riesgo de perder la solución óptima real.

Una forma de reducir el esfuerzo computacional asociado y aumentar la velocidad de convergencia es con la aplicación de una nueva heurística, mediante la cual se limita el número de estrategias guardadas en cada etapa a una cantidad menor o igual a un número predefinido S_N.

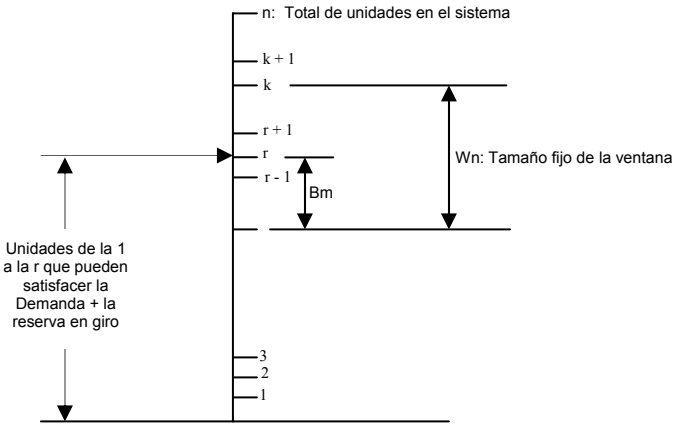


Ilustración de la ventana fija de búsqueda en Programación Dinámica. [2]

La selección de las S_N estrategias se hace eligiendo aquellas que presentan los menores costos asociados. Los costos asociados con una estrategia, los cuales representan una transición de una combinación valida en la etapa *"previa"* a una combinación válida en el etapa *"actual"*, es calculada por la siguiente fórmula:

$$Cost(j,k) = \underset{l}{Min}\left\{Fc(j,k) + Tc(j-1,l:j,k) + Cost(j-1,l)\right\}$$

donde:

- *Cost(j, k)* corresponde al valor del costo asociado con la estrategia *k* en la etapa *j*.
- *Fc(j, k)* corresponde al costo de combustible de la combinación *k* en la etapa *j*.
- *Tc(j – 1, l : j, k)* corresponde al costo de la transición de la combinación *l* en la etapa *j – 1*, a la combinación *k* en la etapa *j*.

En resumen, para hacer factible la aplicación de la Programación Dinámica en la resolución de problemas de Unit Commitment de gran escala, se deben emplear algoritmos para reducir el espacio de soluciones utilizando la heurística. No obstante, al reducir los espacios de soluciones se obtienen mejores rendimientos en cuanto a tiempos de procesamiento, se corre el riesgo de alejar la solución encontrada del óptimo real del problema.

Además de presentar complicaciones para resolver problemas de gran tamaño y depender de la heurística para su resolución la Programación Dinámica además presenta los siguientes limitaciones para resolver problemas de Unit Commitment [1]:

- No considera costos de parada.
- Los costos de partida deben ser fijos, es decir, deben ser independientes del tiempo que la unidad se mantuvo fuera de servicio.

3.3.3 Unit Decommitment

El Unit Decommitment corresponde a un método para resolver el problema de Unit Commitment basado en un procedimiento denominado *"decommitment"* que corresponde a la desasignación de las unidades presentes en el predespacho. El presespacho es inicializado considerando a todas las unidades que se encuentran disponibles durante el período de estudio. Dicho predespacho inicial típicamente presenta un exceso de reserva en giro lo cual conlleva una operación del sistema no económica dado los costos extras en que se está incurriendo. Para conseguir una operación económica, algunas unidades deberían ser consideradas para salir de operación por una parte o todo el período de estudio. La selección de la unidad candidata a ser desasignada se efectúa mediante un proceso denominado *"el ordenamiento óptimo del Unit Decommitment"*, el cual consiste en la evaluación de las unidades mediante algún índice económico relativo, de esta manera la unidad que resulta con el peor desempeño será la unidad que será extraída del predespacho inicial. Este proceso se efectúa en forma iterativa hasta conseguir el parque generador que cumpla con los requerimientos de demanda y reserva en giro del sistema a mínimo costo.

Lo que caracteriza al método de Unit Decommitment es que el costo total decrece en forma monótonamente en cada iteración, y la solución es siempre factible con respecto a la reserva en giro.

Para explicar con mayor detalle el procedimiento empleado por el método de Unit Decommitment se mostrará una formulación simplificada del problema de unit commitment y en base a ella se mostrará la formulación del Unit Decommitment.

3.3.3.1 Formulación del problema

La función objetivo representa la minimización de los costos totales del sistema:

$$\min\left\{\sum_t\sum_i\left[C_{it}\left(P_{it}\right)\cdot u_{it}+S_{it}\right]\right\} \qquad (11)$$

Además se define la siguiente restricción del balance de carga:

$$\sum P_{it}\cdot u_{it}-D_t=0 \qquad t=1,2,\ldots,T \qquad (12)$$

La función que será minimizada corresponde a la siguiente función objetivo aumentada *(P)*:

$$(P) \qquad Min\left\{\sum_t\left[\sum_i\left[C_{it}\left(P_{it}\right)\cdot u_{it}+S_{it}\right]+\lambda_t\left(\sum_i P_{it}\cdot u_{it}-D\right)\right]\right\} \qquad (13)$$

sujeto a:

- Exceso de reserva positivo:

$$EXS_t = \sum R_{it} \cdot u_{it} - D_t - R_t^{req} \geq 0 \qquad (14)$$

- Costos de partida y detención representados por S_{it}

- Otras restricciones de las unidades, incluyendo límites mínimos y máximos de generación, tiempos mínimos de operación y detención, tasa de toma de carga de unidades, operación forzada de unidades, etc. Por simplicidad estas restricciones no serán incluidas durante la formulación del método.

La notación utilizada en las ecuaciones anteriores son detalladas a continuación:

- C_{it} – costo de operación de la unidad *i* en la etapa *t*.
- P_{it} – generación de la unidad *i* en la hora *t*.
- u_{it} – variable de decisión de la unidad *i* en la etapa *t* (1: ON, 0: OFF).
- S_{it} – costo de partida o detención de la unidad *i* en la hora *t*.
- D_t – carga del sistema en la hora *t*.
- EXS_t – exceso de reserva en giro del sistema en la hora *t*.
- R_{it} – capacidad en giro de la unidad *i* en la etapa *t*.
- R_t^{req} – requerimiento de reserva en giro del sistema en la hora *t*.

3.3.3.2 Formulación del Unit Decomitmment

Supongamos que se tiene la siguiente solución inicial al problema planteado anteriormente $\left(u_{it}^0, P_{it}^0, S_{it}^0, \lambda_t^0\right)$ con un exceso de reserva en giro en el período de estudio. Si se considera un valor fijo para $\lambda_t = \lambda_t^0$, el problema *(P)* es aditiva y separable para el índice *i*, es decir por unidad. Entonces tomando como variable de decisión la variable u_{it}, se formula el siguiente subproblema para cada unidad *i*:

$$(P_i) \qquad \min\left\{\sum\left[C_{it}\left(P_{it}^0\right)\cdot u_{it} + S_{it} - \lambda_t^0 P_{it}^0 \cdot u_{it}\right]\right\} \qquad i = 1,\ldots,I \qquad (15)$$

sujeto a todas las restricciones locales de la unidad i y de reserva en giro del sistema;

$$EXS_t = \sum_{j \neq i} R_{jt} \cdot u_{jt}^0 + R_{it} \cdot u_{it} + D_t - R_t^{req} \geq 0 \qquad (16)$$

El problema (P_i) puede ser resuelto por medio de Programación Dinámica para cada unidad en el subconjunto de unidades restantes. Una característica distintiva de este método es que si una unidad *i* está on-line en la hora *t* en la iteración *"previa"*, dicha unidad es desasignada en la iteración *"actual"* sólo si la restricción (16) es cumplida. Por lo tanto, la factibilidad de la reserva esta siempre garantizada en el proceso de desasignación (decommitment). Entre todas las unidades optimizadas en el problema (P_i), sólo una es seleccionada para ser desasignada en cada iteración. El criterio para seleccionar la unidad que será desasignada en cada iteración del proceso se muestra a continuación.

3.3.3.3 Criterio de desasignación de unidades (Decomitmment)

Para seleccionar la unidad que será desasignada en cada iteración se utiliza el *ahorro relativo de costos* como índice económico. Esto significa que una unidad será desasignada si y sólo si el ahorro relativo de costo es positivo. El proceso de desasignación se detiene cuando el ahorro relativo de costos de todas las unidades es cero. Esta característica garantiza que los costos totales sean monótonamente decrecientes con cada iteración y la convergencia absoluta del proceso. En [7] se muestran dos criterios para realizar esta desasignación, los cuales se resumen a continuación.

El primer criterio corresponde a la determinación de un índice que representa el ahorro relativo de costo de la unidad

Ambos criterios son similares y se basan en el ahorro relativo de costos producto de la desasignación de una unidad, el primero representa el ahorro de costos para la **unidad** *i*, mediante la utilización del subproblema (P_i) y el segundo representa el ahorro de costos para el **sistema**, mediante la utilización del problema (P).

Para determinar el ahorro absoluto de costos para la unidad o para el sistema se debe calcular la diferencia entre los costos totales antes y después de desasignar una determinada unidad, en el primer caso utilizando el problema (P_i) y en el segundo caso el problema (P).

En ambos criterios, el ahorro relativo de costos se calcula como el cuociente entre el ahorro absoluto de costos y el mínimo entre el exceso de reserva en giro del sistema y la generación total pérdida producto de la desasignación de la unidad.

Una vez obtenidos los ahorros relativos de costos de cada unidad, el criterio para decidir qué unidad será desasignada para los dos criterios es el siguiente:

La unidad con el ahorro relativo de costos más alto es seleccionada para ser desasignada en la iteración actual.

Se puede mostrar que el segundo criterio entrega el orden óptimo de desasignación de unidades, sin embargo, este criterio consume mucho más tiempo de procesamiento dado que después de desasignar cada unidad el programa necesita ejecutar un despacho económico para encontrar los λ_t y P_{it} resultantes luego de realizar la desasignación. Esto significa que si hay n unidades candidatas participando en la iteración en curso, entonces se requiere realizar n despachos económicos. Por otra parte, para el primer criterio sólo se requiere efectuar un despacho económico en cada iteración. Según [7], casos generales de prueba han demostrado que ambos criterios entregan casi el mismo orden de desasignación.

3.3.4 Método de Relajación Lagrangeana

3.3.4.1 Planteamiento General

Para realizar la formulación matemática y explicar el procedimiento iterativo utilizado en el algoritmo de relajación lagrangeana, en primer lugar se debe plantear el problema de optimización que se desea resolver. Consideremos entonces el siguiente problema:

$$\varphi = \min c \cdot x \qquad (17)$$

s.a.

$$Ax \le b \qquad (18)$$

$$Bx \le d \qquad (19)$$

$$x_i \text{ son enteros}, i \in I. \qquad (20)$$

donde los vectores:

$b \in \Re^{m_0}$, $d \in \Re^{m_1}$ y $c \in \Re^n$

y las matrices:

$A \in \Re^{m_0 \times n}$ y $B \in \Re^{m_1 \times n}$

En la formulación anterior las restricciones del problema son divididas en dos grupos (18) y (19), esta separación en el caso de Unit Commitment corresponde a la siguiente:

- Las restricciones representadas por (18) corresponderán a las restricciones de acoplamiento del problema (restricciones "complejas")
- Las restricciones representadas por (19) corresponderán a las restricciones individuales (restricciones "simples").

Se llama relajación Lagrangeana del problema de optimización definido por (17) – (20) a la siguiente formulación:

$$\begin{aligned} &\psi(\lambda) = \min \{c \cdot x + \lambda \cdot (b - Ax)\} \\ &s.a. \\ &Bx \le d \\ &x_i \text{ son enteros}, i \in I \end{aligned} \qquad (21)$$

donde $\lambda \in \Re_+^{m_0}$ es un vector de multiplicadores de Lagrange no negativos.

La formulación descrita en (21) tiene sentido cuando resolver dicho problema es mucho más sencillo que resolver el problema original de optimización ((17) – (20)).

En el caso de Unit Commitment lo anterior se cumple pues con la formulación descrita en (21), las restricciones (19) corresponden a restricciones no acopladas entre sí y por lo tanto, el problema puede ser resuelto mediante una descomposición en donde para cada unidad generadora se resuelva un subproblema independiente, los cuales se resuelven en forma separada. el problema de optimización puede ser resuelto mediante optimizaciones parciales.

El método de la Relajación Lagrangeana se fundamenta en una importante propiedad dual [4]:

$$\psi(\lambda) \leq \varphi$$

donde:

$\psi(\lambda)$ corresponde a la solución óptima de (21) para $\lambda \in \Re_+^{m_0}$

φ corresponde a la solución óptima del problema descrito por (17) – (20).

Lo anterior indica que la solución del la relajación Lagrangeana (21) corresponderá siempre a una cota inferior del problema primal (17) – (20). Producto de lo anterior se hace importante determinar cual es el valor de λ que entrega la mejor estimación de la cota inferior. Lógicamente, la mejor cota inferior estará dada por la mayor de ellas, de está forma surge la necesidad de definir el siguiente problema de optimización denominado "problema dual de Lagrange":

$$\begin{aligned} & z = \max \psi(\lambda) \\ & s.a.: \lambda \in \Re_+^{m_0} \end{aligned} \qquad (22)$$

Dicho problema, al ser $\psi(\lambda)$ una función convexa y lineal por partes, puede resolverse mediante cualquiera de los abundantes métodos de optimización de problemas convexos no diferenciables que existen en la literatura.

La cantidad $\Delta = z - \varphi$ es llamada la "duality grap" (brecha dual) del problema definido por (17) – (20) con respecto ala restricción (18). El menor valor de Δ corresponderá a la estimación más exacta que puede obtenerse mediante la relajación Lagrangeana del problema. Dicha "duality grap" aparece dada la existencia de variables enteras y la no convexidad del dominio admisible del problema (17) – (20). De esta forma, el valor de la "duality grap" caracteriza el grado de "no convexidad" del problema.

3.3.4.2 Aplicación en Unit Commitment

La idea básica de la relajación Lagrangeana es relajar las restricciones asociadas a la demanda y reserva requeridas por el sistema, utilizando para ello multiplicadores de Lagrange y formando una estructura jerarquica de optimización. El nivel inferior de optimización consta de subproblemas individuales, uno para cada unidad generadora, mientras el nivel más alto corresponde a la optimización de los multiplicadores para resolver el problema dual. Ambos niveles son resueltos en forma iterativa.

Para la formulación del método de Relajación Lagrangeana para resolver el problema de Unit Commitment, es necesario clasificar las restricciones en dos grupos, tal como se mostró en el punto anterior, es decir, restricciones de acoplamiento y restricciones individuales. Asimismo, se considerará que la función objetivo estará comprendida por la suma de los costos de operación, que incluye los costos de combustible, y los costos de partida y detención de las máquinas.

Examinando la función objetivo planteada en (1), se puede observar que esta es separable por unidad, es decir, corresponde a la suma de los costos de cada unidad asociados a su operación. A

partir de este precepto se desarrolla el método denominado relajación Lagrangeana, el cual toma ventaja de esta característica y define multiplicadores de lagrange λ^t asociados con las restricciones de balance energético (2) y multiplicadores de lagrange μ^t asociados con las restricciones de capacidad de reserva en giro (3) y reserva en giro del sistema (4). Mediante dichos multiplicadores, estas restricciones se agregan a la función objetivo como penalidades generando de esta forma la función de Lagrange *L*:

$$L = F - \sum_{t}^{T}\left[\lambda^t \cdot \left(\sum_{n}^{N} U_{n,t} \cdot P_{n,t} - D_t\right) - \mu_1^t \cdot \left(\sum_{n}^{N} U_{n,t} \cdot \left(P_{n,t}^{\max} - P_{n,t}\right) - R^t\right) - \mu_2^t \cdot \left(\sum_{n}^{N} U_{n,t} \cdot Rmáx_n - R^t\right)\right]$$

Donde F corresponde a la función a minimizar en la función objetivo indicada en (1).

$$F = \sum_{t}^{T}\sum_{n}^{N}\left(U_{n,t} \cdot FC(P_{n,t}) + S_{n,t}^{start} + S_{n,t}^{stop}\right)$$

Considerando que los costos de detención corresponden a un valor fijo para cada unidad, este término se puede extraer de la función objetivo, con lo cual se obtiene la siguiente función objetivo simplificada:

$$F = \sum_{t}^{T}\sum_{n}^{N}\left(U_{n,t} \cdot FC(P_{n,t}) + S_{n,t}^{start}\right)$$

Por otra parte, podemos definir la función q_n de manera que se cumpla lo siguiente:

$$L = \sum_{n}^{N} q_n + K$$

donde:

$$q_n = \min_{P_{n,t}, U_{n,t}} \left\{\sum_{t}^{T}\left[\left(U_{n,t} \cdot FC(P_{n,t}) + S_{n,t}^{Start}\right)\right] - \sum_{t}^{T}\lambda^t \cdot \left(U_{n,t} \cdot P_{n,t}\right) - \sum_{t}^{T}\left(\mu_1^t \cdot U_{n,t} \cdot P_{n,t}^{máx} + \mu_2^t \cdot U_{n,t} \cdot Rmáx_n\right)\right\}$$

$$K = \sum_{t}^{T}\left[\lambda^t \cdot D_t + \mu_1^t \cdot \left(D_t + R^t\right) + \mu_2^t \cdot R^t\right]$$

El problema q_n se resuelve mediante Programación Dinámica para cada unidad.

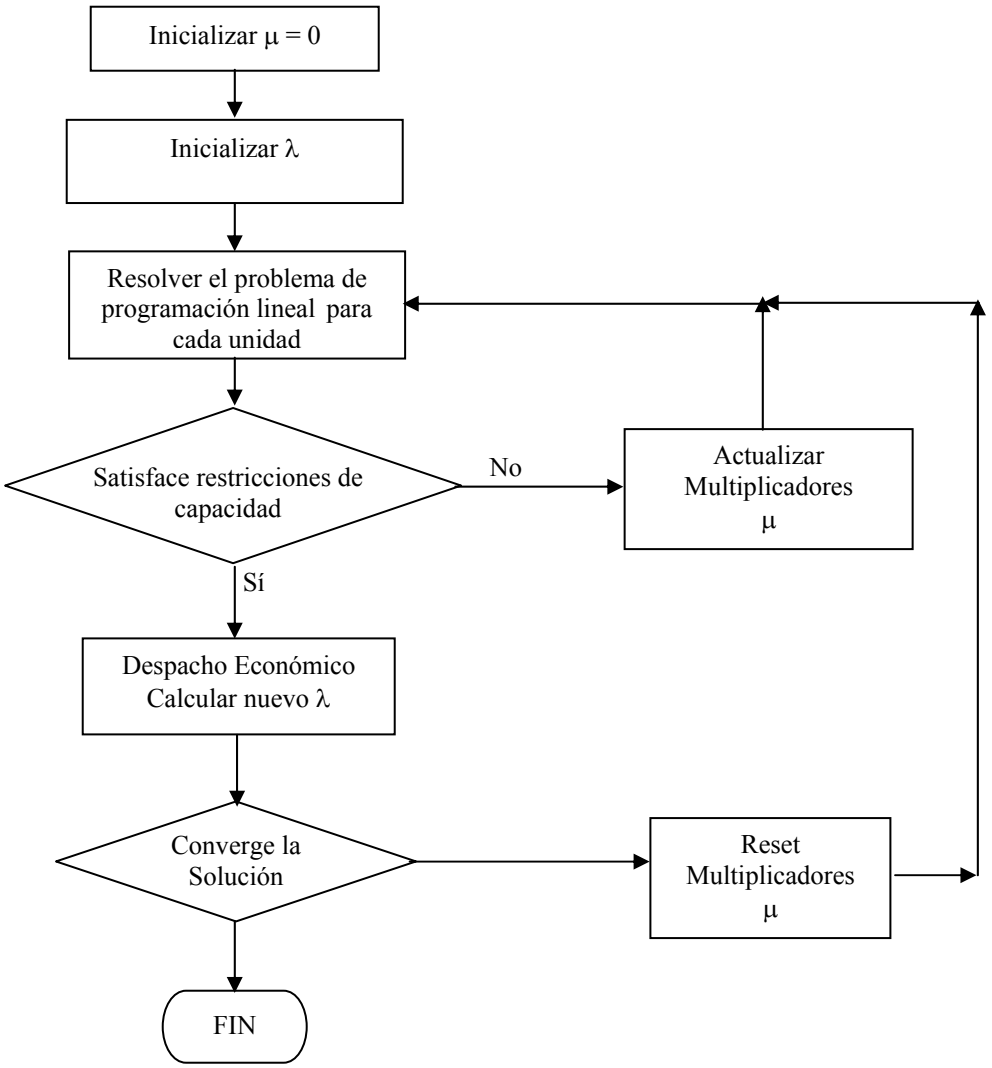
Para seleccionar los valores asociados a los multiplicadores de lagrange para cada iteración se resuelve el siguiente problema dual de Lagrange.

$$\underset{\lambda,\mu}{máx}\left\{\sum_{n}^{N} q_n + \sum_{t}^{T}\left[\lambda^t \cdot D_t + \mu_1^t \cdot \left(D_t + R^t\right) + \mu_2^t \cdot R^t\right]\right\}$$

El proceso iterativo general para resolver el problema de Unit Commitment por relajación Lagrangeana es el siguiente:

1. Una buena estimación inicial para λ puede ser obtenida olvidando las restricciones que dan la dependencia del tiempo. Los multiplicadores μ son inicializados en cero.
2. Plantear y resolver los problemas de programación dinámica para cada unidad (q_n) considerando el conjunto de restricciones individuales planteadas en 1.3.2. Con esto se obtienen $(P_{n,t}, U_{n,t}), \forall\, n,t$.
3. Verificar si la solución encontrada en el punto anterior satisface la restricción de balance energético. Si no la satisface pasar al punto 4, por el contrario, pasar al punto 5.
4. Actualizar los multiplicadores resolviendo el problema dual de Lagrange. Volver al punto 2.
5. Manteniendo fijos los $U_{n,t}$ calculados se resuelve un despacho económico para determinar nuevos valores para los multiplicadores λ.
6. Para determinar la convergencia se puede determinar la brecha dual mediante la diferencia entre el resultado de las funciones objetivo del primal y dual.
7. Si no se ha alcanzado la convergencia, se inicializar los multiplicadores μ en cero y se vuelve al punto 2.

El proceso anterior se resume en la siguiente figura:



3.3.4.3 Consideraciones

La relajación Lagrangeana es un muy buen método de aproximación para resolver el problema de Unit Commitment. La ventaja más obvia de este método es su eficiencia computacional en especial para sistemas de gran tamaño. Si bien la heurística esta usualmente involucrada en la obtención de despachos factibles, la calidad de los despachos puede ser cuantitativamente evaluada.

Por otra parte, los multiplicadores de Lagrange llamados precios sombra tienen un significado económico muy claro. En un ambiente de mercado esta es una importante ventaja. Los multiplicadores son los precios de mercado de la energía y las reservas.

La desventaja más obvia de este método es que los despachos obtenidos en la solución dual son generalmente infactibles. La heurística podría ser requerida para generar despachos factibles. Mientras mayor es el número de restricciones complejas, tales como: restricciones de transmisión u otras restricciones de seguridad son consideradas, más difícil se hace obtener soluciones factibles basadas en heurística. Si consideramos un ambiente de mercado competitivo, las soluciones cercanas al óptimo podrían afectar negativamente resultados de mercado premiando a unidades de baja eficiencia en perjuicio de máquinas más eficientes. Esto podría resultar en un perjuicio sistemático contra algunos participantes del mercado dependiendo de la heurística involucrada en obtener soluciones factibles.

3.3.5 Métodos Generales de Programación Entero Mixta

Existen dos principales métodos para resolver problemas de programación entero mixta: método *cutting planes* y *Branch and Bound*.

La idea básica del método *cutting planes* es resolver un problema de programación lineal (LP) cuya región factible contenga todas las soluciones factibles del problema original. El problema LP tiene la misma función objetivo del problema original y es usualmente llamado una relajación LP. Si la solución de la relajación LP satisface las restricciones del problema original, ésta debería ser la solución óptima del problema de programación entera mixta. Si lo anterior no ocurre, una nueva restricción de desigualdad (llamada *cutting plane*) es introducida, de manera de extraer la solución fraccional de la región factible del problema de relajación LP en curso. Una nueva relajación LP es entonces formada.

El método de *Branch and Bound* se basa en un concepto distinto. El número de los posibles valores del vector binario que determina la componente entera del problema original es finito, si este vector binario es determinado, el problema original se transforma en un problema LP que es fácil de resolver. La solución óptima se puede obtener si todos los posibles problemas LP son resueltos. Sin embargo esta exhaustiva enumeración normalmente es imposible dado el gran número de posibilidades. Por ello el número de problemas LP a ser resueltos es reducido mediante la organización en estructura de árbol de los problemas LP y definiendo una cota inferior para la función de costo en cada rama. Si la cota inferior obtenida en una rama es peor que la mejor solución factible obtenida hasta ese momento, esta rama es "*podada*". De esta forma

sólo un pequeño porcentaje de problemas LP es resuelto y la solución óptima es eventualmente obtenida.

A continuación se muestran en detalle los algoritmos de los dos principales métodos empleados para resolver problemas de programación entero mixta: Método de Benders y Branch and Bound.

3.3.5.1 Método de Benders

De la misma manera que en método de la Relajación Lagrangeana se efectuaba una separación entre las restricciones, clasificándolas en "complejas" y "simples" (para el caso de Unit Commitment restricciones de acoplamiento e individuales respectivamente), en el método de Benders se realiza una partición respecto a las variable, separándolas en variables discretas (complejas) y variables continuas (simples).

Una descripción general del método consiste en que en cada iteración se fijan las variables discretas y se resuelve un problema auxiliar continuo. La información obtenida al resolver este problema es utilizada para configurar un problema mediante el cual se ajustan los valores de las variables complejas.

Para realizar la formulación del método de Benders consideraremos el siguiente problema de optimización entero-mixta:

$$\varphi = \min\{f(x) + cy\} \qquad (1)$$

s.a.

$$G(x) + Ay \le b \qquad (2)$$

$$x \in S,\ y \in \Re^n \qquad (3)$$

donde $S \subset Z^{n_1}$ es un conjunto finito de enteros para los cuales la función f y el vector de función G, de dimensión m, están definidas; c y b son vectores de dimensión n y m respectivamente. Además A es una matriz de dimensión m x n.

El problema definido por (1) – (3) corresponde a un problema de programación entero mixta, en el cual las restricciones y la funcional son lineales con respecto a las variables continuas (y) .

Consideremos la siguiente función $\pi(x)$ con dominio S:

$$\pi(x) = \min\ \{cy \mid A \le b - G(x)\ \} \qquad (4)$$

Para un valor fijo de $x \in S$, la función definida en (4) es un problema de programación lineal. El dual de este problema queda definido por [6]:

$$\pi'(x) = \max\ \{\lambda(b - G(x)) \mid \lambda A = c,\ \lambda \in \Re_+^m\} \qquad (5)$$

Para el problema anterior el conjunto de soluciones factibles corresponde al siguiente poliedro convexo:

$$\Lambda_1 = \left\{\lambda \mid \lambda A = c,\ \lambda \in \Re_+^m\right\} \qquad (6)$$

Se puede inferir que el problema (1) es equivalente respecto de su funcional[1] y de las variables[2] discretas al siguiente problema:

$$\varphi = \min\left\{f(x) + \pi(x) \mid x \in S\right\} \qquad (7)$$

Analizando el conjunto de soluciones factibles Λ_1 se puede inferir lo siguiente:

- Si $\Lambda_1 = \phi$, entonces el problema (5) no tiene solución, es decir, $\pi'(x) = +\infty$, por lo tanto en virtud de la dualidad de la programación lineal $\pi(x) = -\infty$ y por consiguiente $\varphi = -\infty$, es decir el problema primal no tiene solución.

- Si $\Lambda_1 \neq \phi$, entonces $\pi'(x) = \pi(x)$ para todo $x \in S$.

Para el segundo caso se tiene que el problema (7) es equivalente con respecto a su funcional y las variables discretas al siguiente problema:

$$\varphi = \min\left\{f(x) + \pi'(x) \mid x \in S\right\} \qquad (8)$$

Introduciendo la variable continua θ se puede rescribir el problema anterior de la siguiente forma:

$$\varphi = \min_{x,\theta}\left\{f(x) + \theta \mid \theta \geq \pi'(x),\ x \in S\right\} \qquad (9)$$

Se define el conjunto Λ_2 de la siguiente forma:

$$\Lambda_2 = \left\{\lambda \in \Re^m \mid \exists\, \lambda' \in \Re^m,\ \lambda' + q\lambda \in \Lambda_1,\ q \geq 0\right\}$$

donde Λ_1 corresponde al conjunto definido en (6).

Para continuar con la descripción del método de Benders es necesario plantear el siguiente lema:

Lema 1:
$\pi'(x) = +\infty$ si y sólo si, existe un vector $\lambda' \in \Lambda_2$ tal que $\lambda'(b - G(x)) > 0$.

La demostración del lema anterior se puede encontrar en [6].

---

[1] Definición de equivalencia respecto de la funcional en Anexo 1: Definiciones.

[2] Definición de equivalencia respecto de las variables en el Anexo 1: Definiciones.

Por el Lema 1 se tiene que, el conjunto $\{x \mid \pi'(x) < +\infty\}$ es definido por el siguiente sistema de restricciones:

$$\lambda(b - G(x)) \leq 0, \quad \lambda \in \Lambda_2$$

El par (θ, x) es factible con respecto a las restricciones del problema (9) si y sólo si:

$$\theta \geq \lambda(b - G(x)), \quad \lambda \in \Lambda_1$$

Con lo anterior el problema (9) puede ser reescrito como:

$$\varphi = \min_{x,\theta} \{f(x) + \theta\} \qquad (10)$$

s.a.

$$\theta \geq \lambda(b - G(x)), \quad \lambda \in \Lambda_1 \qquad (11)$$

$$0 \geq \lambda(b - G(x)), \quad \lambda \in \Lambda_2 \qquad (12)$$

$$x \in S \qquad (13)$$

En general. El problema definido por (10) – (13) tiene infinitas restricciones. En el método de Benders se utiliza una aproximación por iteración en un esquema denominado "relajación de restricciones". Consiste en resolver en cada iteración una versión acotada de (10) – (13) por un conjunto limitado de restricciones. Esta variante suele ser llamada problema coordinador o maestro de Benders. Luego se resuelve el problema auxiliar de programación lineal que determine la restricción más excedida o violada del tipo (11) o una restricción excedida del tipo (12). La restricción obtenida es incluida en el problema maestro y el proceso es repetido. La primera solución del problema maestro, factible con respecto a todas las restricciones (11) y (12), será la óptima solución del problema (10) – (13).

Descripción Formal del Algoritmo de Solución de Benders

En primer lugar, se definen los conjuntos $L_1 \subset \Lambda_1$ y $L_2 \subset \Lambda_2$ mediante los cuales se definirá para cada iteración el subconjunto acotado de restricciones del problema maestro. A dichos conjuntos se les llama los cortes de Benders que forman las restricciones de tipo (11) y (12) respectivamente.

Para la primera iteración se considera $L_1 = \phi$ y $L_2 = \phi$ y se agrega una restricción del tipo $\theta \geq M$, con M lo suficientemente pequeño, de manera de garantizar que el problema maestro tenga solución.

Para cada iteración *k, k = 1, 2, ...* se realiza el siguiente procedimiento:

1. Resolver el problema maestro de Benders de la iteración *k*:

$$\begin{cases} r(k) = \min\limits_{x,\theta} \{f(x) + \theta\} \\ \theta \geq \lambda(b - G(x)),\ \lambda \in L_1 \\ 0 \geq \lambda(b - G(x)),\ \lambda \in L_2 \\ x \in S \end{cases} \qquad (14)$$

   Sean $(\theta^{(k)}, x^{(k)})$ la solución óptima del problema anterior. Donde $L_1 \subset \Lambda_1$ y $L_2 \subset \Lambda_2$, entonces se cumple que $r(k) \leq \varphi$.

2. Resolver el problema auxiliar de programación lineal definido en (5). Con ello se obtienen dos posibles casos:

   a. $\pi'(x^{(k)}) = +\infty$. Entonces, resolviendo el problema (5) con $x = x^{(k)}$ mediante algún método de programación lineal, se encuentra el vector $\lambda^{(k)} \in \Lambda_2$ tal que $\lambda^{(k)}(b - G(x^{(k)})) > 0$. Entonces $\lambda^{(k)}$ es incluido en el conjunto L_2 y se pasa a la iteración siguiente. De esta forma se obtiene una restricción del tipo (12), la cual es violada para $\theta = \theta^{(k)}$ y $x = x^{(k)}$.

   b. $\pi'(x^{(k)}) < +\infty$. Sea $\lambda^{(k)}$ la solución óptima del problema definido en (5), con $x = x^{(k)}$. Entonces, la restricción $\theta \leq \lambda^{(k)}(b - G(x))$ es la "más excedida" del conjunto de restricciones definidas en (2), para $\theta = \theta^{(k)}$ y $x = x^{(k)}$ (las restricciones definidas por (12) son satisfechas en virtud de que $\pi'(x^{(k)}) < +\infty$).

   Si la diferencia $\theta^{(k)} - \lambda^{(k)}(b - G(x^{(k)}))$ es igual a cero, entonces $(\theta^{(k)}, x^{(k)})$ es una solución factible de del problema definido por (10) – (13) (y por lo tanto una solución optima por la construcción del problema maestro).

   Si $\theta^{(k)} - \lambda^{(k)}(b - G(x^{(k)})) < 0$, entonces $\lambda^{(k)}$ es incluido en el conjunto L_1 y se pasa a la iteración siguiente.

   De esta forma se completa la descripción del procedimiento para cada iteración.

   En cada iteración se obtiene la cota inferior $r(k)$ y una cota superior $f(x^{(k)}) + \pi'(x^{(k)})$ para el valor óptimo φ de la funcional del problema (1) – (3), las cotas inferiores son monótonamente crecientes:

$$r(1) \leq r(2) \leq r(3) \leq \ldots \leq r(k) \leq r(k+1) \leq \ldots$$

3.3.5.2 Branch and Bound

El método de Branch and Bound corresponde a un algoritmo para resolver problemas de programación entero mixta.

Este método se caracteriza por resolver problemas en donde las variables enteras se relajan considerándose como continuas y por lo tanto se generan problemas relajados de programación lineal (relajación LP) que puede ser resuelto por algún algoritmo de programación lineal de mayor simplicidad. De la solución encontrada en el problema relajado (nodo) se generan divisiones en el espacio de soluciones factibles generando subproblemas (ramas) con conjuntos disjuntos de búsqueda. A estos conjuntos se les aplica el mismo procedimiento de división, de esta manera se genera un árbol de búsqueda.

Con las soluciones obtenidas en cada nodo del árbol de búsqueda se determina aquella "dominante" que representará la cota superior (para problemas de maximización) del problema original, la cual será utilizada para determinar las ramas que irán siendo podadas (finalizadas) durante el proceso de generación del árbol.

Este proceso termina cuando se han recorrido todas las soluciones factibles o cuando se encuentra una solución óptima de un problema relajado que pertenece al conjunto de soluciones factibles del problema original, en cuyo caso dicha solución será la óptima.

A continuación se describen con mayor detalle: el algoritmo general de Branch and Bound y el método de Branch and Bound usando relajación LP.

I. Algoritmo general de Branch and Bound

Consideremos el siguiente problema de programación entera:

$$(IP) \qquad z_{IP} = máx\{cx: \quad x \in S\}$$

Se define $\Lambda = \{IP^i\}$ como el conjunto de problemas de programación entera en donde cada uno es da la forma $z_{IP}^i = máx\{cx: \quad x \in S^i\}$ donde $S^i \subseteq S$. Asociado con cada problema de Λ existe una cota superior $\bar{z}^i \geq z_{IP}^i$.

Algoritmo:

1º. *Inicialización*: $\Lambda = \{IP\}$, $S^0 = S$, $\bar{z}^0 = \infty$, y $\hat{z}_{IP} = -\infty$ (solución "dominante").

2º. *Prueba de término*: Si $\Lambda = \phi$, entonces la solución x^0 que da $\hat{z}_{IP} = cx^0$ es la solución óptima.

3º. *Problema de selección y relajación*: Seleccionar y borrar un problema IP^i de Λ. Resolver el problema relajado RP^i. Considerando que z_R^i corresponde al valor óptimo del problema relajado y x_R^i la solución óptima de dicho problema, si es que la solución existe.

4º. *Poda*:

   a. Si $z_R^i \leq \hat{z}_{IP}$, ir al punto 1.

   b. Si $x_R^i \notin S^i$, ir al punto 5.

   c. Si $x_R^i \in S^i$ y $cx_R^i > \hat{z}_{IP}$, entonces se actualiza el valor de la solución "dominante", $\hat{z}_{IP} = cx_R^i$. Borrar de Λ todos los problemas cuya cota superior sea inferior o igual a la solución "dominante", es decir, $\bar{z}^i \leq \hat{z}_{IP}$. Si $cx_R^i = z_R^i$, ir al punto 2, sino, ir al punto 5.

5º. *División*: Sea $\{S^{ij}\}_{j=1}^k$ una división de S^i. Se agregan a Λ los problemas $\{IP^{ij}\}_{j=1}^k$ asociados a la división anterior, donde $\bar{z}^{ij} = z_R^i$ para $j = 1, \ldots, k$. Ir al punto 2.

II. Branch and Bound Usando Relajación LP

A continuación se presentará el algoritmo Branch and Bound que utiliza una relajación sobre programación lineal.

Se considerará el problema general de programación entera siguiente:

$$(IP) \qquad z_{IP} = máx\{cx: \quad x \in S\}$$

donde:

$$S = \{x \in Z_+^n: \quad Ax \leq b\}$$

La relajación inicial, S es reemplazado por $S_{LP}^0 = \{x \in R_+^n: \quad Ax \leq b\}$ y el problema relajado $z_R(x) = cx$ para cada relajación.

Criterio para la Poda

Supongamos la siguiente relajación LP en el nodo *i* del árbol de enumeración:

$$(LP^i) \qquad z_{LP}^i = máx\left\{cx: \quad x \in S_{LP}^i\right\}, \text{ donde } S_{LP}^i = \left\{x \in R_+^n: \quad A^i x \le b^i\right\}.$$

Si LP^i tiene una solución óptima, ésta se denotará por x^i. Luego, las condiciones para la poda[3] serán las siguientes:

1. $S_{LP}^i = \phi$: Criterio de infactibilidad, es decir, no existen soluciones factibles para el subproblema.
2. $x^i \in Z_+^n$: Criterio de optimalidad, se encontró una solución óptima que pertenece al conjunto de soluciones óptimas del problema de programación entera.
3. $z_{LP}^i \le \hat{z}_{IP}$, donde $\hat{z}_{IP}$ es el valor de una solución factible del problema de programación entera que es conocido (solución "dominante"). Otra forma de aplicación de este criterio es mediante la aplicación de una condición denominada "débil", esto es $z_{LP}^i \le \hat{z}_{IP} + \xi$ para alguna tolerancia ξ dada.

Las proposiciones que sustentan estos criterios y su demostración correspondiente se encuentran descritas en la referencia [1].

División

Debido a que se resuelve una relajación LP en cada nodo, ésta da origen a una división del espacio de soluciones factibles cuando el óptimo del problema relajado no pertenece al conjunto de soluciones factibles del problema de programación entera. Una forma de generar estas divisiones corresponde a tomar $S = S^1 \cup S^2$ con $S^1 = S \cap \left\{x \in R_+^n: \quad dx \le d_0\right\}$ y $S^2 = S \cap \left\{x \in R_+^n: \quad dx \ge d_0 + 1\right\}$, donde $d \in Z^n$ y $d_0 \in Z$. Si x^0 es la solución de la relajación:

$$(LP^0) \qquad z_{LP}^0 = máx\left\{cx: \quad x \in R_+^n,\ Ax \le b\right\},$$

se pueden seleccionar (d, d_0) tal que, $d_0 < dx^0 < d_0 + 1$, de esta forma $x^0 \notin S_{LP}^1 \cup S_{LP}^2$ y por lo tanto, $z_{LP}^i = máx\left\{cx: \quad x \in S_{LP}^i\right\} < z_{LP}^0$.

Una forma utilizada para seleccionar (d, d_0) es la denominada "*Dicotomía Variable*", esta consiste en lo siguiente:

Supongamos que la solución x^0 de la relajación es infactible debido a que la variable $x_j^0 \notin Z^1$ para algún $j \in N$, en ese caso d corresponde al vector canónico e_j y $d_0 = \left\{\overline{d} \in Z: \quad x_j^0 - 1 < \overline{d} < x_j^0\right\}$.

---

[3] Se llama poda a la acción de finalizar la ramificación o generación de subproblemas de una determinada rama del árbol de enumeración.

Selección de Nodo

Dada una lista Λ de subproblemas activos o, equivalentemente, un árbol parcial de nodos activos o que no han sido podados, se debe determinar cual será el nodo siguiente que será examinado en detalle. Para ello existen dos opciones básicas: (1) Reglas por lista de prioridad, mediante la cual se determina el orden en que el árbol será desarrollado, (2) Regla adaptiva que selecciona un nodo utilizando información del estado (cotas, etc.) de los nodos activos.

Selección de la variable que genera el árbol o "Branching variable"

Supongamos que se ha seleccionado un nodo activo i , el cual tiene asociada la solución al problema de programación lineal $x^i = \left(x_1^i, x_2^i, \ldots, x_n^i\right)$ y se define $N^i = \left\{j \in N : \quad x_j^i \notin Z^1\right\}$ como el conjunto de índices de las variables que hacen infactible la solución para el problema de programación entera. Entonces se debe seleccionar el índice $j \in N^i$ para el cual se generará el árbol (divisiones del espacio de soluciones factibles). Evidencia empírica muestra que la selección de $j \in N^i$ puede ser muy importante para los tiempos de ejecución del algoritmo.

Una forma común de seleccionar la *Branching variable* es mediante una *prioridad definida por usuario*, esto es, un ordenamiento de las variables es especificado como parte de los datos de entrada y las *branching variables* son seleccionadas de N^i de acuerdo a este orden.

Otras posibilidades involucran la *degradación* o *penalización*. Ambas intentan estimar el decrecimiento en $\bar{z}^i$ causado por el requerimiento de que x_j^i sea entero. El criterio general consiste en seleccionar la variable que presente la máxima degradación.

3.3.5.3 Consideraciones

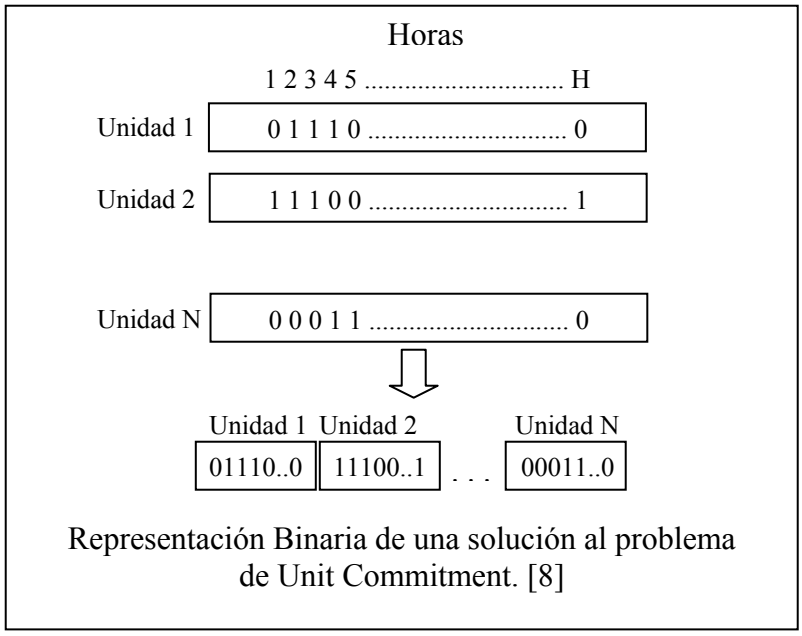
La ventaja más obvia de los métodos generales de resolución de problemas de programación entero mixta corresponde a que no requieren heurística para determinar los despachos óptimos, por lo que agregar más restricciones al problema es simple y no conlleva esfuerzos significativos para cambiar el algoritmo. Por otra parte, la principal desventaja de estos métodos sigue siendo su gran complejidad computacional, la cual aumenta mientras más grande es el sistema analizado.

3.3.6 Algoritmos Genéticos

Los Algoritmos Genéticos corresponden a técnicas de búsqueda basados en principios inspirados de la genética y mecanismos de evolución observados en sistemas naturales y en poblaciones de organismos vivientes. Su principio básico es el mantener una población de soluciones para un problema (genotipos), como información codificada y de manera individual que evoluciona en el tiempo. La evolución está basada en las leyes de selección natural y recombinación de información genética dentro de la población. La evolución de la población muestra el espacio de búsqueda y acumula conocimiento acerca de las buenas y malas cualidades, recombinando este conocimiento para formar soluciones con óptimo desempeño para un problema específico.

3.3.6.1 Unit Commitment utilizando Algoritmos Genéticos

Para la aplicación del algoritmo genético al UC, se utiliza una codificación binaria para representar cada solución (genotipo). Si se tienen N unidades y H etapas horarias en el horizonte de estudio, entonces asumiendo que en cada hora los estados posibles para una unidad son ON o OFF, se requiere de un arreglo de H bit para representar el predespacho de una unidad en cada solución. En dicho arreglo, un "1" en una determinada ubicación significa que la unidad se encuentra en el estado ON en esa hora, mientras que un valor "0" representa que la unidad se encuentra en el estado "OFF". De acuerdo a lo anterior, al concatenar los arreglos de N unidades se conformará un arreglo de NxH bit, como se muestra en la figura 1.



Representación Binaria de una solución al problema de Unit Commitment. [8]

En primer lugar, se generan en forma aleatoria M soluciones iniciales (genotipos) que conforman la población inicial. Luego cada genotipo es evaluado mediante una función predeterminada que mide de cierta forma la calidad de cada solución, para ello utiliza parámetros tales como, costo total de operación, número total de partidas y salidas de servicios de unidades, número de restricciones violadas y cantidad en que cada restricción fue violada. Dichos parámetros son

multiplicados por multiplicadores de penalización de manera conveniente. En [8] se propone la siguiente función para evaluar cada genotipo:

$$F = FC_T + SU_T + SD_T + \sum_{j=1}^{NC} PF_j$$

$$con \quad PF_j = \mu_j \cdot \left| VIOL_j \right|$$

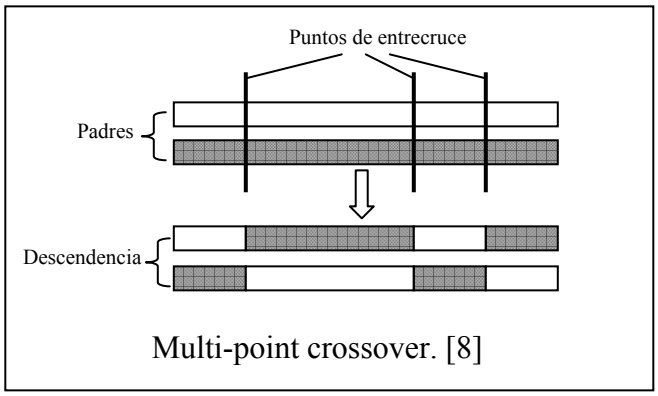
donde:

FC_T	: Costo total de combustible
SU_T (SD_T)	: Costo total asociado a partidas (salidas) de unidades
NC	: Número de restricciones de operación violadas.
$VIOL_j$	: Cantidad en que fue violada la restricción j.
PF_j	: Penalización asociada con la violación de la restricción j.
μ_j	: Multiplicador de penalización asociado con la restricción j.

Luego de la evaluación de la población inicial, el algoritmo genético comienza la creación de la nueva generación, para ello se realiza una selección de los padres (pares) mediante un algoritmo que selecciona los genotipos con una probabilidad inversamente proporcional al valor obtenido en su evaluación. Dos padres darán origen a dos genotipos nuevos, mediante una recombinación de sus bit de acuerdo a los siguientes operadores:

a. Entrecruce (crossover)

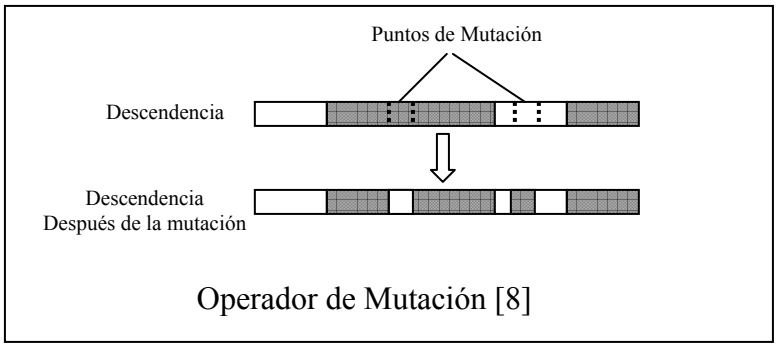
Este operador es aplicado con una cierta probabilidad. Cuando los genotipos de los padres son combinados (intercambio de bit) para formar dos nuevos genotipos que heredan las características de ambos padres. Una forma es la denominada *"multi-point crossover"*, la cual se muestra en la figura 2.



Multi-point crossover. [8]

b. Mutación

Posee una pequeña probabilidad de aplicación. Bits son seleccionados en forma aleatoria de una descendencia y son cambiados de "0" a "1" o viceversa, como muestra la figura 3.



Operador de Mutación [8]

Los procedimientos anteriores son repetidos hasta obtener M nuevos genotipos los cuales son considerados como una nueva generación de soluciones. La nueva generación reemplaza completamente a sus padres.

En el proceso de evolución también son incluidas algunas otras características como lo son copiar la mejor solución de cada generación en la generación siguiente (elitismo), la función de evaluación de cada genotipo es escalada por una transformación no lineal, en orden a enfatizar las pequeñas diferencias entre cualidades cercanas al óptimo en una población de convergencia (fitness scaling) y otros.

Este proceso es repetido en forma iterativa obteniendo en cada iteración una nueva generación de soluciones. Los criterios utilizados para detener el proceso usualmente son un número máximo de iteraciones o ausencia de mejoras en nuevas generaciones.

Este proceso de selección encuentra soluciones cercanas al óptimo con muy alta probabilidad, sin embargo adolece de dos problemas fundamentales: requiere largos tiempos de procesamiento computacional para encontrar una solución cercana al óptimo y que requiere una rigidez de las restricciones a lo largo de la evolución del programa.

4 PREDESPACHO UTILIZANDO RELAJACIÓN LAGRANGEANA

El algoritmo de Realajación Lagrangeana deriva su nombre de la conocida técnica matemática que utiliza multiplicadores de Lagrange para resolver problemas de optimización con restricciones, pero es en realidad una técnica de descomposición utilizada para resolver problemas de programación matemática de gran escala. La metodología de la relajación lagrangeana utiliza una descomposición que genera subproblemas de predespacho para cada unidad generadora en forma independiente en el horizonte de planificación.

4.1 El Problema de Optimización

Función Objetivo

El problema de predespacho para un período de tiempo T en un sistema con N unidades, presenta la siguiente función objetivo:

$$F = \min \left\{ \sum_{t=1}^{T} \sum_{n=1}^{N} \left(F_n\left(P_{n,t}\right) \cdot u_{n,t} + S_{n,t}^{start} + S_{n,t}^{stop} \right) \right\} \tag{1}$$

La cual corresponde a la suma de los costos operativos más los costos de partida y los costos de detención.

Restricciones de acoplamiento

- *Balance Energético*

$$\sum_{n=1}^{N} u_{n,t} \cdot P_{n,t} = D_t \qquad \forall\, t = 1,\ldots,T \tag{2}$$

D_t : Demanda bruta del sistema en el tiempo *t*

- *Capacidad de Reserva en Giro*

$$\sum_{n=1}^{N} u_{n,t} \cdot \left(P_{n,t}^{\max} - P_{n,t}\right) \geq R^t \qquad \forall\, t = 1,\ldots,T \tag{3}$$

Restricciones Individuales

- *Restricciones de Capacidad*

$$P_{n,t}^{\min} \le P_{n,t} \le P_{n,t}^{\max} \qquad \forall\, n = 1,\ldots,\, N \quad \forall\, t = 1,\ldots,\, T \tag{4}$$

- *Capacidad de reserva en giro de cada unidad*

$$P_{n,t}^{\max} - P_{n,t} \ge R_{n,t} \qquad \forall\, n = 1,...,N \quad \forall\, t = 1,...,\, T \tag{5}$$

Donde Rn,t corresponde a la reserva en giro requerida para la unidad n en la hora t.

- *Límites de tiempo*

$$u_{n,t} \quad \begin{cases} 1 & si \quad t_{n,t}^{on} \le \hat{t}_n^{on} \\ 0 & si \quad t_{n,t}^{off} \le \hat{t}_n^{off} \end{cases} \forall\, n = 1,\ldots,\, N \quad \forall\, t = 1,\ldots,\, T \tag{6}$$

4.2 Relajación Lagrangeana

La función de Lagrange asociada al problema de optimización planteado anteriormente corresponde a la siguiente expresión:

$$L = F - \sum_{t=1}^{T}\left[\lambda^t \cdot \left(\sum_{n=1}^{N} u_{n,t} \cdot P_{n,t} - D_t\right) - \mu_t \cdot \left(\sum_{n=1}^{N} u_{n,t} \cdot \left(P_{n,t}^{\max} - P_{n,t}\right) - R^t\right)\right] \tag{7}$$

donde:

$$F = \sum_{t=1}^{T}\sum_{n=1}^{N}\left(u_{n,t} \cdot FC(P_{n,t}) + S_{n,t}^{start} + S_{n,t}^{stop}\right) \tag{8}$$

Luego la relajación lagrangeana del problema primal corresponderá a la minimización de la función lagrangeana sujeta a las restricciones individuales presentadas anteriormente.

Si se considera a los multiplicadores como valores fijos, la minimización de la función lagrangeana se puede separar en N subproblemas de optimización independientes, uno por cada unidad generadora.

$$L = \sum_{n=1}^{N} q_n + k \tag{9}$$

donde:

$$q_n = \sum_{t=1}^{T} \left[u_{n,t} \cdot FC(P_{n,t}) + S_{n,t}^{start} + S_{n,t}^{stop} - \left(\lambda^t \cdot P_{n,t} \cdot u_{n,t} + \mu_t \cdot (P_{n,t} - P_{n,t}^{\max}) \cdot u_{n,t} \right) \right] \tag{10}$$

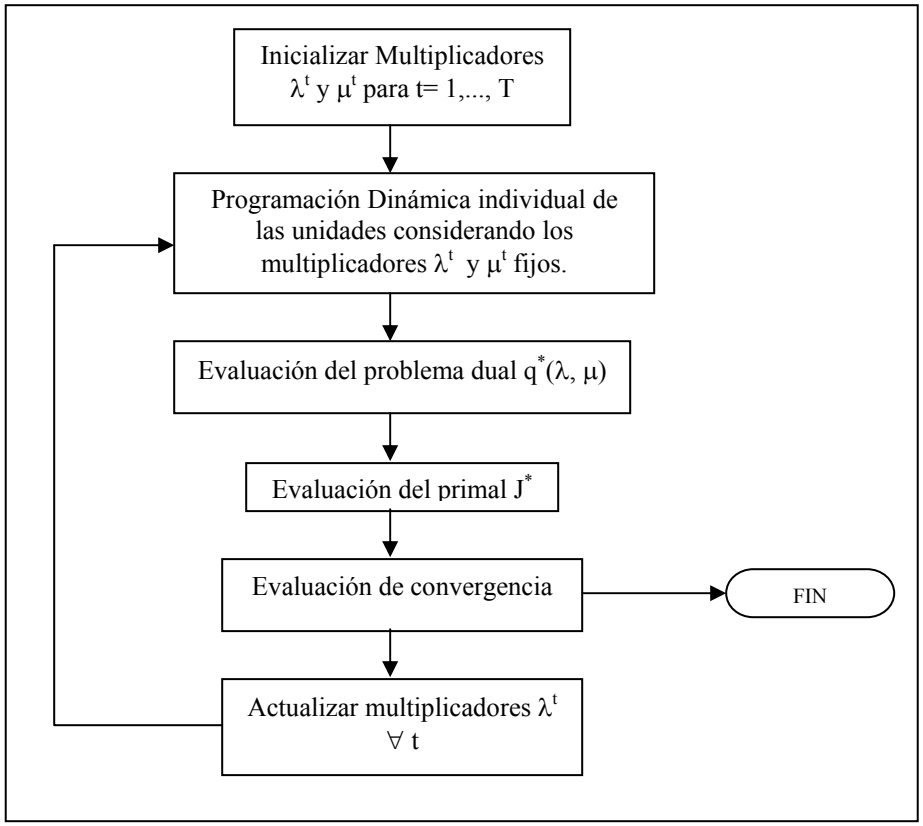
y k es una constante definida por la siguiente expresión:

$$k = \sum_{t=1}^{T} \left(\lambda^t \cdot D_t - \mu_t \cdot R^t \right) \tag{11}$$

Con la descomposición anterior se tiene que la minimización de la función lagrangeana L cuando se consideran los multiplicadores fijos es equivalente a la minimización de las funciones qn para $n = 1,..., N$, sujeto a las restricciones individuales (4), (5) y (6). Estos problemas son de fácil solución mediante programación dinámica.

4.3 Algoritmo de Predespacho

En la siguiente figura se muestra un esquema que representa las etapas que conforman el algoritmo de relajación lagrangeana para resolver el problema de predespacho.



A continuación se presenta una descripción de cada una de las etapas indicadas en el diagrama anterior.

4.3.1 Inicialización de los multiplicadores de Lagrange

En esta etapa se deben encontrar valores para los multiplicadores de Lagrange, los cuales serán considerados como fijos para la etapa siguiente del algoritmo que corresponde a la programación dinámica individual de las unidades.

Dicha selección puede realizarse de distintas formas y generalmente se requiere de la utilización de heurística para determinar los valores más adecuados para obtener un buen resultado en la optimización.

Los multiplicadores asociados a la restricción de balance energético λt son interpretados económicamente como el precio de la generación de un kWh adicional de energía en el sistema

en el instante de tiempo t, por lo cual usualmente estos multiplicadores se inicializan en valores que representan en forma aproximada este valor.

Una forma de obtener una buena estimación para estos multiplicadores es mediante la determinación horaria de un despacho que cumpla con los requerimientos de demanda mediante la asignación de unidades por orden de mérito y sin considerar las restricciones dependientes del tiempo, de esta forma el valor para cada hora de dichos multiplicadores corresponderá al costo marginal de la unidad más cara presente en el despacho.

Por otra parte los multiplicadores asociados a las restricciones de requerimientos de reserva generalmente se asocian al precio del siguiente kW de reserva en giro en un instante de tiempo. Este valor se puede aproximar por la siguiente expresión:

$$\left(\beta_k - \lambda^t\right) \cdot \frac{P_k^{\min}}{P_k^{\max}} \qquad (12)$$

donde β_k es el costo marginal de la unidad k necesaria para cumplir con la reserva en giro mínima exigida y λ_t es el multiplicador de lagrange asociado a la restricción de balance energético en el instante de tiempo t. La deducción de esta expresión se puede observar en los anexos sección B.

Luego los multiplicadores asociados a la restricción de reserva pueden ser inicializados de acuerdo al siguiente criterio:

$$\mu_t = 0 \qquad \text{si } \sum_{n=1}^{N} u_{n,t} \cdot \left(P_{n,t}^{\max} - P_{n,t}\right) \geq R^t$$

$$\mu_t = \left(\beta_k - \lambda^t\right) \cdot \frac{P_{k,t}^{\min}}{P_{k,t}^{\max}} \qquad \text{si } \sum_{n=1}^{N} u_{n,t} \cdot \left(P_{n,t}^{\max} - P_{n,t}\right) < R^t$$

4.3.2 Programación Dinámica Individual de las Unidades

En esta etapa del algoritmo se resuelven los problemas de optimización individual de cada unidad mediante programación dinámica. En esta etapa los multiplicadores de lagrange se consideran fijos y se deben cumplir las siguientes restricciones:

- Limites operativos de las unidades (4)
- Capacidad de reserva en giro de cada unidad (5)
- Tiempos mínimos de detención y operación (6).

Mediante este proceso de optimización se obtienen valores para las potencias de cada unidad en cada instante de tiempo y para las variables enteras de decisión.

El algoritmo empleado para la aplicación de la programación dinámica individual de las unidades se encuentra detallado en los anexos sección C.

4.3.3 Evaluación del Problema Dual

La evaluación del problema dual consiste en sustituir en la función de lagrange el valor fijo de los multiplicadores de lagrange λ_t y μ_t, el vector binario de decisión u_t y las potencias obtenidas por programación dinámica individual de las unidades P_t. Mediante esta evaluación se obtiene el valor de la función dual $q^*(\lambda,\mu)$.

4.3.4 Evaluación del Primal

Para evaluar el primal (J*) es necesario encontrar una solución factible al problema del predespacho completo. La manera de encontrar esta solución factible puede ser utilizando técnicas basadas en heurísticas o bien mediante la aplicación de despachos económicos horarios basados en el vector binario del dual. Esta última técnica consiste en sustituir las variables binarias (enteras) obtenidas de la programación dinámica individual de las unidades en el problema original. Al asumir conocidas las variables enteras, el problema resultante se resuelve por programación lineal o cuadrática.

El planteamiento general del problema de despacho económico y una reseña de las principales técnicas utilizadas para resolverlo se detallan en los anexos sección E.

Para la efectuar la evaluación del primal en la programación, se utilizará el despacho económico basado en el vector binario del dual, resolviendo el despacho económico mediante la aplicación del método del subgradiente (Anexos sección D).

4.3.5 Verificación de convergencia

La verificación de la convergencia del método se realiza evaluando la diferencia relativa entre la solución del primal y del problema dual, la cual es denominada comúnmente en la literatura como "relative duality gap" y se define por la siguiente expresión:

$$RDG = \frac{J^* - q(\lambda,\mu)}{q(\lambda,\mu)}$$

Luego, el criterio empleado para detener el algoritmo corresponderá a $RDG \le \varepsilon$ para un ε suficientemente pequeño.

4.3.6 Actualización de los Multiplicadores de Lagrange

Esta etapa corresponde a la maximización de la función Lagrangeana respecto de las variables duales (λ_t, μ_t), las cuales deben cumplir con la condición de no negatividad.

Para resolver este problema de maximización se utilizará el método del subgradiente (ver Anexos sección D), es decir,

$$\lambda^{(k+1)} = \lambda^{(k)} + \alpha \cdot \frac{\partial q(\lambda)}{\partial \lambda}$$

$$\mu^{(k+1)} = \mu^{(k)} + \beta \cdot \frac{\partial q(\mu)}{\partial \mu}$$

Donde α y β son las longitudes de paso y corresponden a parámetros de sincronización del modelo que serán determinados durante el proceso de programación.

5 PREDESPACHO UTILIZANDO ALGORITMO BRANCH AND BOUND

El algoritmo Branch and Bound se utiliza para resolver problemas de programación entero mixta y por lo tanto es una técnica útil para resolver problemas de predespacho de unidades generadoras.

Este algoritmo se fundamenta en la generación de un árbol de búsqueda, el cual tiene como raíz el problema original de programación entera relajado, es decir, las variables enteras se consideran continuas, obteniendo así un problema mucho más sencillo y que puede ser resuelto con un algoritmo de mayor simplicidad. A partir de este nodo raíz se generan subproblemas que irán conformando las ramas del árbol, a los cuales se les agregarán progresivamente restricciones de igualdad de manera de generar soluciones en espacios de búsqueda disjuntos que lleven a obtener una solución factible para el problema original de programación entera.

Con las soluciones factibles, para el problema original de programación entera, que se vayan obteniendo en la medida que se recorre cada nodo del árbol se determina aquella solución que entregue el menor costo, la cual corresponderá a la solución "dominante", luego, cualquier solución de un subproblema que entregue una solución cuyo costo sea mayor al de la solución "dominante" será "podado", es decir, no se continuará la búsqueda en los descendientes de dicho nodo. Lo anterior debido a que las soluciones de los hijos de un determinado nodo siempre serán mayores (en el caso de una minimización) que la solución del nodo padre y por ende de la solución dominante.

En este esquema los nodos del espacio de búsqueda se pueden etiquetar de tres formas distintas: *nodo vivo*, *muerto* o *en expansión*.

Un *nodo vivo* es un nodo factible y prometedor del que no se han generado todos sus hijos, por otra parte, un *nodo muerto* es un nodo del que no van a generarse más hijos por alguna de las tres razones siguientes:

- ya se han generado todos sus hijos o
- no es factible o
- no es prometedor (un nodo es prometedor si la información que tenemos de ese nodo indica que expandiéndolo se puede conseguir una solución mejor que la mejor solución en curso).

En cualquier instante del algoritmo pueden existir muchos nodos vivos y muchos nodos muertos pero sólo existe un nodo en expansión que es aquél del que se están generando sus hijos en ese instante.

Claramente, la búsqueda termina cuando todos los nodos han sido finalizados, ya sea porque la rama a la que pertenecen ha sido podada o porque se ha encontrado una solución factible al problema original. La solución óptima por consiguiente corresponderá a la solución dominante obtenida al final del proceso.

Existen múltiples formas de aplicar el algoritmo Branch and Bound para resolver el problema de predespacho, las cuales dependerán de la forma en que se defina la relajación inicial y la forma de recorrer el árbol de búsqueda.

El árbol de búsqueda puede ser recorrido de acuerdo a alguno de los siguientes procedimientos: *vuelta atrás* o por *ramificación y poda*. El primero corresponde a una búsqueda ciega mientras que el segundo es una búsqueda informada, luego, la diferencia es el orden en que se recorren los nodos del espacio de búsqueda.

- ***Vuelta Atrás***: es una búsqueda ciega, lo cual significa que fijado un nodo x del espacio de búsqueda el siguiente nodo a visitar es el primer hijo de x sin visitar, en un recorrido en profundidad y el siguiente hermano de x, en un recorrido en anchura. Tan pronto como se genera un nuevo hijo del nodo en curso, este hijo se convierte en el nuevo nodo en curso o nodo en expansión y los únicos nodos vivos son los que se encuentran en el camino que va desde la raíz al actual nodo en expansión.

- ***Ramificación y Poda***: es una búsqueda informada, en la cual, fijado un nodo x del espacio de búsqueda el siguiente nodo es el más prometedor de entre todos los nodos vivos; es el que se va a convertir en el próximo nodo en expansión. Todos los hijos del nodo en expansión se generan antes de que cualquier otro nodo vivo pase a ser el nuevo nodo en expansión, esto implica que se debe conservar en algún lugar muchos más nodos vivos que pertenecen a distintos caminos. Ramificación y Poda utiliza una lista para almacenar y manipular los nodos vivos.

A continuación se presenta la forma propuesta en la presente memoria para resolver el problema de predespacho utilizando el algoritmo Branch and Bound.

5.1 Algoritmo de predespacho

El problema de predespacho original corresponde al mismo presentado en la sección 4.1 y para el presente algoritmo será denominado como “problema maestro de programación entera (PMPE)”.

En primer lugar se debe definir el problema original relajado (P.O.R.) que determinará la solución inicial y la cota inferior del problema.

La herramienta de optimización utilizada corresponde a la función de Matlab denominada *fmincon*, la cual corresponde a una función que minimiza funciones de varias variables, continuas, con restricciones del tipo lineales y no lineales.

Considerando las limitaciones de la herramienta de optimización utilizada, se estableció que el P.O.R. debía tener la menor complejidad posible, es por ello que se determinó relajar las variables enteras y las restricciones asociadas al cumplimiento de los tiempos mínimos de operación y detención. Cabe señalar que esta última relajación no se agregará como restricción adicional a los subproblemas asociados a cada nodo del árbol de búsqueda sino que se exigirá al momento de realizar la evaluación de factibilidad de cada subproblema, es decir, al agregar una restricción a alguna variable de decisión (ON/OFF) se analizará si dicha condición impuesta es factible, en lo relativo a las restricciones de tiempo, considerando como dato las restricciones impuestas en los nodos padres.

Luego el planteamiento del P.O.R. es el siguiente:

$$F = \min \left\{ \sum_{t=1}^{T} \sum_{n=1}^{N} \left(F_n\left(P_{n,t}\right) \cdot u_{n,t} + S_{n,t}^{start} + S_{n,t}^{stop} \right) \right\}$$

Sujeto a:

$$\sum_{n=1}^{N} u_{n,t} \cdot P_{n,t} = D_t \qquad \forall\, t = 1,\ldots,T$$

$$\sum_{n=1}^{N} u_{n,t} \cdot \left(P_{n,t}^{\max} - P_{n,t}\right) \geq R^t \qquad \forall\, t = 1,\ldots,T$$

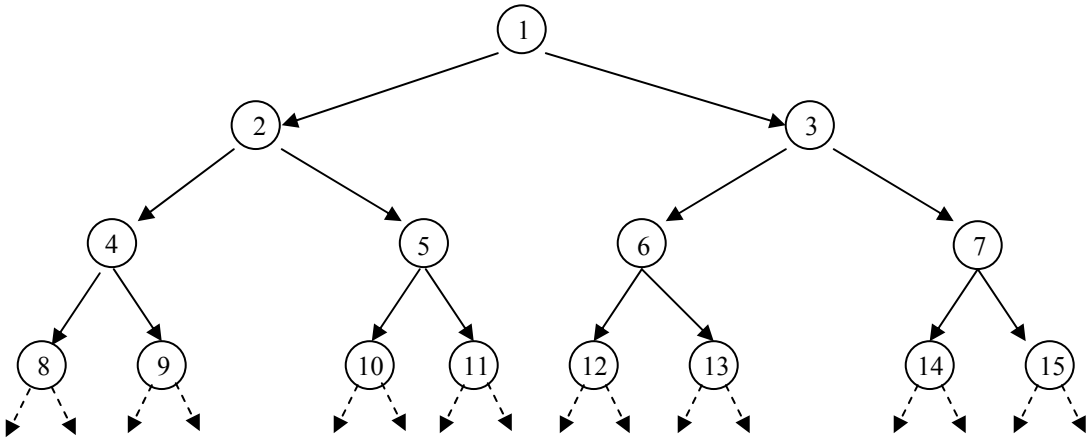
$$P_{n,t}^{\min} \leq P_{n,t} \leq P_{n,t}^{\max} \qquad \forall\, n = 1,\ldots,N \quad \forall\, t = 1,\ldots,T$$

$$0 \leq u_{n,t} \leq 1 \qquad \forall\, n = 1,\ldots,N \quad \forall\, t = 1,\ldots,T$$

Cada nodo del árbol de búsqueda corresponde a un subproblema de optimización determinado por el P.O.R. al cual se le agrega un conjunto R de restricciones. Dichas restricciones corresponden a asignar un valor binario a alguna de las variables de dedición $u_{n,t}$. De esta forma para un nodo cualquiera x, el primer hijo corresponderá al subproblema determinado por el problema asociado al nodo padre, al cual se le agregará la restricción $\{u_{n,t} = 1\}$, luego el segundo hijo del nodo x se determinará de la misma forma pero la restricción a agregar corresponderá a $\{u_{n,t} = 0\}$.

Dado que el P.O.R. no considera las restricciones de tiempos mínimos de operación ni detención de las unidades, cada vez que se genera un nuevo nodo y por ende se agrega una restricción, se debe realizar un análisis de factibilidad, de manera de verificar que con la nueva restricción agregada, se pueda obtener una solución que cumpla con las condiciones de tiempo de cada unidad, considerando como dato las restricciones heredadas por los padres del nodo analizado.

La numeración elegida para los nodos, que no representa el orden en que serán recorridos, se muestra en la siguiente figura:



De esta forma el primer hijo del nodo 1 es el nodo 2 y el segundo es el nodo 3 y en general, el primer hijo del nodo *p* será el nodo $2 \cdot p$ y el segundo el nodo $2 \cdot p + 1$.

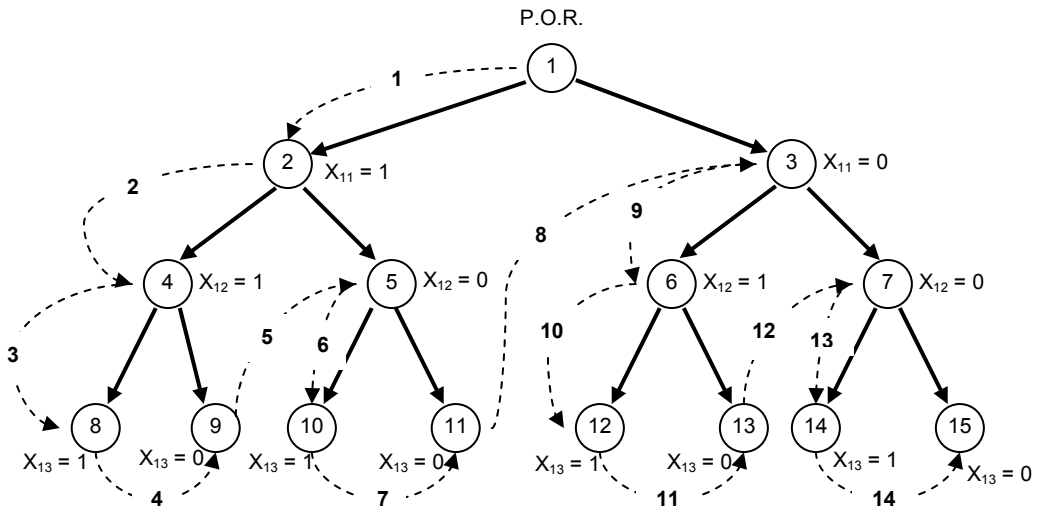
Como se mencionó anteriormente, se considerará que al primer hijo de cada nodo se le asignará una restricción del tipo $\{u_{n,t} = 1\}$. Por otra parte, el esquema elegido para realizar la búsqueda en el árbol de búsqueda corresponderá a *vuelta atrás*, es decir, se realizará un recorrido en profundidad, es decir, el nodo siguiente para un nodo vivo siempre será el primer hijo. Por otra parte cuando se llegue a un nodo muerto se continuará con un recorrido en anchura, es decir, el siguiente nodo para un nodo muerto será un nodo vivo que corresponderá a su hermano o el nodo padre más próximo.

La explicación de la forma en que se recorrerán los nodos en el árbol de búsqueda y de la asignación de restricciones para cada nodo se realizará mediante el siguiente ejemplo.

Si consideramos un sistema de 3 unidades para un horizonte de tiempo igual a 1 etapa, el árbol de soluciones estará compuesto por 15 nodos en total, los cuales serán recorridos de acuerdo al procedimiento descrito anteriormente en el siguiente orden de nodos: 1 – 2 – 4 – 8 – 9 – 5 – 10 – 11 – 3 – 6 – 12 – 13 – 7 – 14 – 15.

En la figura siguiente se muestra esquemáticamente el árbol de soluciones, con la restricción agregada en cada nodo y la ruta seguida por el algoritmo.

Se debe considerar que $X = \{X_{jk}\}_{j=1,\ldots N}^{k=1,\ldots,T}$ es el vector binario de decisión y para este ejemplo N=1 y T=3.



2 Nodos

Indica relación o parentesco entre nodos

Indica ruta con que se recorre el árbol, el número muestra el orden.

$X_{13} = 1$ Restricción agregada al conjunto de restricciones R en cada nodo

Cabe señalar que cada vez que se produce una vuelta atrás, por ejemplo en la ruta número 5 (nodo 9 a nodo 5), la restricción asociada al nodo que *quedó atrás* es extraída del conjunto de restricciones *R* (para el caso del ejemplo, nodo 9 a nodo 5 será: $R - \{X_{13} = 0\}$).

Un nodo entrega una solución factible para el PMPE cuando la solución de la optimización del subproblema asociado corresponde a una matriz de decisión con elementos binarios y se cumplen las condiciones de tiempo mínimo de operación y detención para todas las unidades.

Algoritmo Branch and Bound

Consideremos el siguiente problema de programación entera mixta:

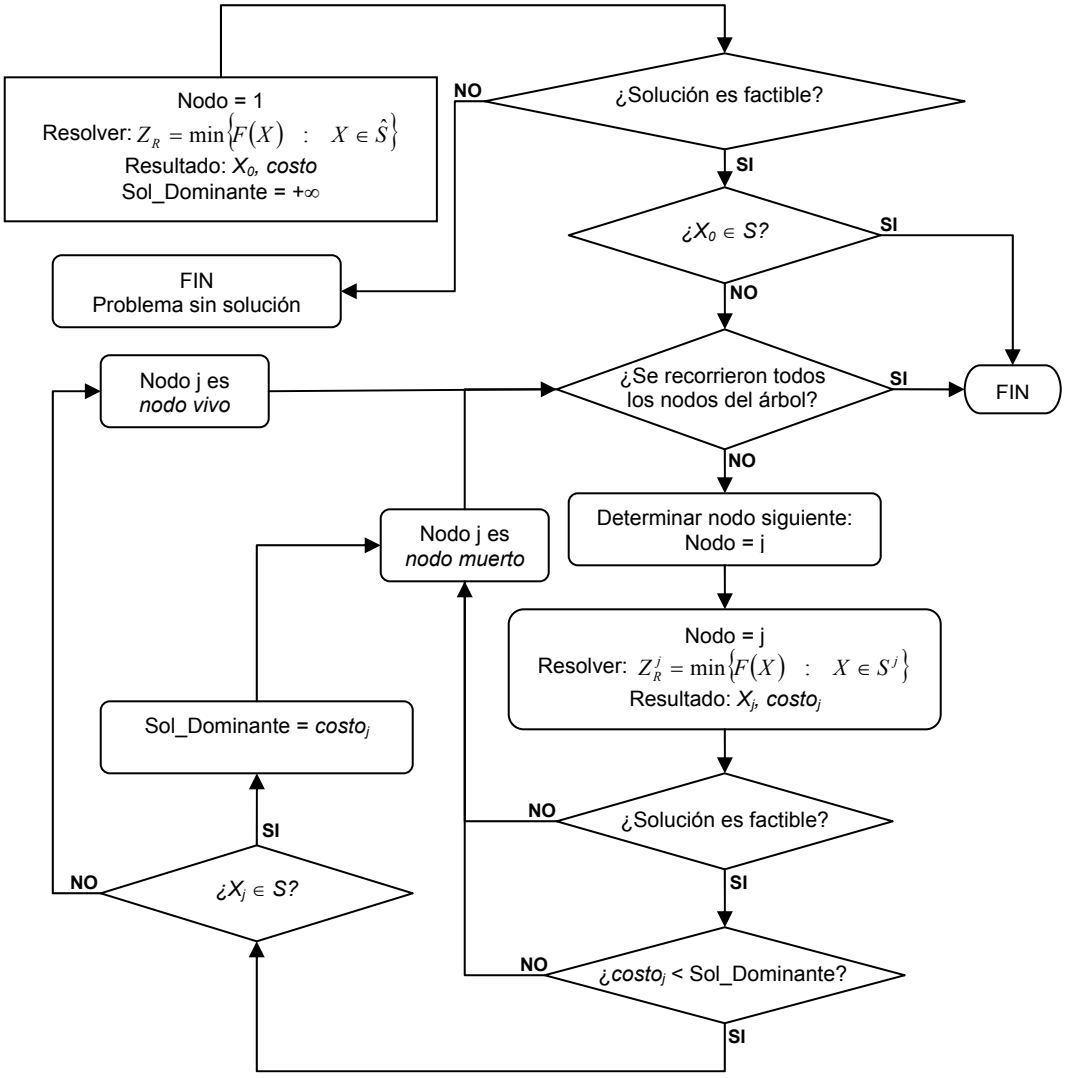
$$Z_{MIP} = \min\{F(X) \ : \ X \in S\}$$

Luego supongamos que $Z_R = \min\{F(X) \ : \ X \in \hat{S}\}$ es el P.O.R. asociado a Z_{MIP}, en donde se cumple que $S \subset \hat{S}$.

Se define S^i como el conjunto de soluciones posibles asociado al nodo *i* del árbol de búsqueda, luego el subproblema asociado al nodo i-ésimo corresponderá a:

$$Z_R^i = \min\{F(X) \ : \ X \in S^i\}, \ S^i \subset \hat{S}.$$

Utilizando la notación anterior, el algoritmo Branch and Bound se puede esquematizar de la siguiente forma.



6 PROGRAMACIÓN DE ALGORITMOS DE PREDESPACHO

Para realizar la evaluación de los algoritmos: Branch and Bound y Relajación Lagrangeana para la optimización de la operación de corto plazo, previamente se requiere la realización de programas que resuelvan mediante estas técnicas el problema de predespacho.

En el presente capitulo se presenta una descripción general de la estructura de cada programa y resultados obtenidos de pruebas preliminares sobre sistemas pequeños.

Los programas se realizaron con en la plataforma MATLAB versión 6.0 y se utilizó como motor de optimización la función del TOOLBOX de MATLAB *FMINCON*. La función *FMINCON* resuelve problemas de minimización de funciones de varias variables con restricciones lineales y no lineales mediante la utilización de un método de Programación Cuadrática Secuencial (SQP). Este método resuelve en cada iteración un subproblema de programación cuadrática. Además en cada iteración se determina la matriz Hessiana, empleando una estimación de los Multiplicadores de Lagrange.

Las pruebas sobre estos programas fueron realizadas en un computador personal con un procesador Pentium IV de 2.4 GHz y 384 MB de memoria RAM.

6.1 Programación del algoritmo de Relajación Lagrangeana

La programación del algoritmo de relajación lagrangeana consta de las siguientes funciones principales:

function Lagrangean_Relaxation()

Programa principal que recibe como datos de entrada los parámetros técnicos de las unidades, los requerimientos de demanda y reserva, las funciones de costo de las unidades y las condiciones iniciales del problema. Entrega como resultado, la matriz de variables de decisión (ON/OFF), las potencias asociadas a las unidades y el costo total del predespacho.

Considera un número máximo de iteraciones y una brecha dual máxima para determinar la convergencia.

function Inicializa_mult1()

Función encargada de determinar los valores iniciales de los multiplicadores λ y μ., para ello efectúa un despacho económico mediante lista de prioridad utilizando como costo asociado a cada unidad el costo medio de operación de la unidad a potencia máxima:

$$CU_{\max}(k) = \frac{C_k(P^k_{\max})}{P^k_{\max}}$$

Donde:

Ck: función de costo de la unidad k.
$P^k_{\max}$: Potencia máxima de la unidad k.

Luego el valor del multiplicador λ en el instante de tiempo t corresponderá al costo medio a potencia de la última unidad despachada para cumplir la demanda del sistema en t.

Por otra parte, el multiplicador μ en un determinado instante de tiempo t se determinará evaluando la reserva en giro total disponible en dicho instante, de esta manera si la reserva disponible es mayor o igual que la reserva requerida, el multiplicador será nulo, de lo contrario el multiplicador quedará determinado por la siguiente expresión:

$$\mu_t = (\beta_t - \lambda_t) \cdot \frac{P_{\min}^k}{P_{\max}^k}$$

Donde βt corresponde al costo medio a mínimo técnico de la unidad que precede a la última unidad despachada según el orden de mérito establecido, λt el multiplicador de lagrange asociado a la restricción de demanda en el instante t y, $P_{\min}^k$ y $P_{\max}^k$ la potencia mínima y máxima de la unidad requerida para cumplir con la reserva respectivamente.

Esta función recibe como parámetros las demandas horarias, las reservas en giro requeridas en cada hora, las potencias máximas y mínimas de las unidades, el número de unidades y el número de etapas del problema. El resultado entregado corresponde a dos vectores con la estimación inicial de los valores horarios de los multiplicadores asociados a las restricciones de demanda y de reserva en giro del sistema.

function FO

Esta función recibe como parámetro de entrada un escalar que representa la potencia de una unidad y calcula el valor de la función objetivo a minimizar en la programación dinámica individual de las unidades.

function Programacion_dinamica ()

Corresponde a la subrutina encargada de efectuar la programación dinámica individual de las unidades. Las variables de entrada principales de esta función corresponden al estado inicial de las unidades (ON/OFF y tiempo inicial en una de estas condiciones de cada unidad), tiempos mínimos de operación y detención de las unidades, multiplicadores de lagrange y los costos de partida y detención de las unidades.

Esta función se encarga de determinar las unidades a despachar en cada etapa del período de estudio mediante la resolución de problemas individuales para cada unidad generadora. La función objetivo a minimizar corresponde a la siguiente:

function Funcion_lagrange ()

Corresponde a la subrutina que evalúa la función dual considerando los multiplicadores de lagrange fijos y el vector binario de decisión y la potencia de las unidades obtenidas de la programación dinámica individual.

function Primal ()

Esta función efectúa la evaluación del problema primal, para cumplir este objetivo se debe encontrar una solución factible al problema original. Para obtener dicha solución factible se resuelve para cada etapa del período de evaluación un despacho económico basado en el vector binario de decisión obtenido de la programación dinámica individual de las unidades. La optimización requerida para resolver cada despacho económico se resuelve mediante la utilización de la función de Matlab denominada fmincon, la cual resuelve problemas de minimización de funciones no lineales de varias variables con restricciones lineales y no lineales.

function act_lambda ()

Esta función calcula una actualización de los multiplicadores asociados a la restricción de demanda utilizando el método del subgradiente con la regla de longitud de paso constante:

$$\alpha_k = h \Big/ \left\| g^{(k)} \right\|_2$$

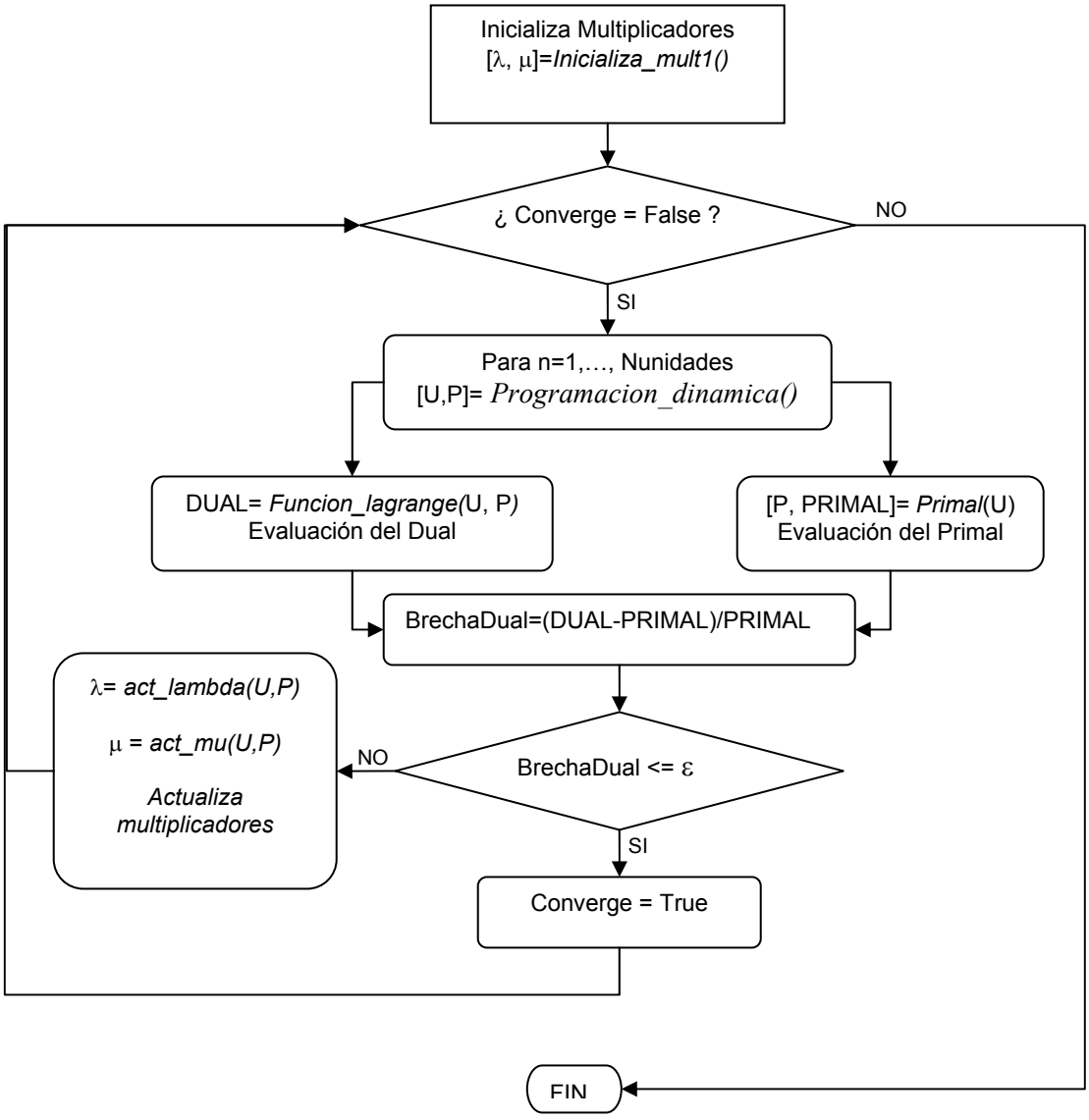
function act_mu ()

Esta función calcula una actualización de los multiplicadores asociados a la restricción de reserva utilizando el método del subgradiente con la regla de longitud de paso de series no sumables pero de cuadrados sumables.

$$\alpha_k = h \Big/ \left\| g^{(k)} \right\|_2$$

6.1.1 Diagrama de flujo del programa de predespacho mediante algoritmo de Relajación Lagrangeana

La estructura general del programa del algoritmo de relajación lagrangeana es el siguiente:



6.1.2 Prueba preliminar del programa

Se efectuaron pruebas mediante la resolución de problemas de predespacho a pequeños sistemas ficticios, a continuación se muestran los resultados obtenidos en una de estas pruebas.

Datos técnicos de las unidades

Nº Unidad	Pmax MW	Pmin MW	Ton horas	Toff horas
1	180	100	8	8
2	150	85	15	10
3	220	150	2	0
4	100	40	12	2
5	120	70	8	2
6	125	65	5	4
7	230	100	5	2
8	120	60	8	15
9	120	70	12	10
10	110	55	15	12
11	170	120	12	12
12	75	35	11	2
13	150	50	5	3
14	35	10	5	0
15	40	20	2	0

Ton: corresponde al tiempo mínimo de operación de las unidades.
Toff: corresponde al tiempo mínimo de detención de las unidades.

Condiciones iniciales

Nº Unidad	ton horas	toff horas
1	8	-
2	15	-
3	2	-
4	12	-
5	8	-
6	5	-
7	5	-
8	-	8
9	12	-
10	15	-
11	-	12
12	-	2
13	5	-
14	-	10
15	-	8

ton: corresponde al número de horas en que la unidad estuvo encendida previo al inicio del período de evaluación.

toff: corresponde al numero de horas en que la unidad estuvo apagada previo al inicio del período de evaluación.

Costo de operación

El costo de operación de las unidades se modela mediante funciones lineales, a continuación se presentan las funciones de costo asociadas a cada unidad

Nº Unidad	FC(P) M$
1	1300+0.02*P
2	1000+0.005*P
3	1180+0.05*P
4	1000+0.09*P
5	1500+0.25*P
6	1500+0.05*P
7	800+0.05*P
8	1250+0.05*P
9	1300+0.05*P
10	1180+0.05*P
11	1180+0.05*P
12	1180+0.1*P
13	600+0.005*P
14	600+0.5*P
15	2300+0.05*P

.

Datos del predespacho

Se considera un período de 24 etapas horarias y no se imponen requerimientos mínimos de reserva. En la tabla siguiente se muestra la demanda horaria del sistema para cada hora:

Etapa	Demanda MW
1	1511
2	1396
3	1401
4	1502
5	1530
6	1548
7	1414
8	1422
9	1463
10	1545
11	1547
12	1516

Etapa	Demanda MW
13	1513
14	1520
15	1544
16	1538
17	1472
18	1487
19	1546
20	1483
21	1468
22	1548
23	1531
24	1430

Resultados

El resultado obtenido para un total de 50 iteraciones es el siguiente:

Costo Total = $ 259.950

Brecha dual = 4,8 %

Tiempo de ejecución: 911,6 segundos

Potencias despachadas:

	Generación en MW											
Unidad/Hra.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	180.0	180.0	180.0	180.0	180.0	180.0	180.0	180.0	180.0	180.0	180.0	180.0
2	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0
3	220.0	204.4	205.1	220.0	220.0	220.0	207.3	208.7	215.5	220.0	220.0	220.0
4	61.0	40.0	40.0	52.0	80.0	98.0	40.0	40.0	40.0	95.0	97.0	66.0
5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	230.0	214.3	215.1	230.0	230.0	230.0	217.3	218.7	225.5	230.0	230.0	230.0
8	120.0	104.3	105.1	120.0	120.0	120.0	107.3	108.7	115.5	120.0	120.0	120.0
9	120.0	104.3	105.1	120.0	120.0	120.0	107.3	108.6	115.5	120.0	120.0	120.0
10	110.0	94.3	95.3	110.0	110.0	110.0	97.3	98.7	105.5	110.0	110.0	110.0
11	170.0	154.3	155.3	170.0	170.0	170.0	157.3	158.7	165.5	170.0	170.0	170.0
12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0
14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Total	**1511**	**1396**	**1401**	**1502**	**1530**	**1548**	**1414**	**1422**	**1463**	**1545**	**1547**	**1516**

	Generación en MW											
Unidad/Hra.	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
1	180.0	180.0	180.0	180.0	180.0	180.0	180.0	180.0	180.0	180.0	180.0	180.0
2	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0
3	220.0	220.0	220.0	220.0	217.3	219.6	220.0	218.9	216.3	220.0	220.0	220.0
4	63.0	70.0	94.0	88.0	40.0	40.0	96.0	40.0	40.0	98.0	81.0	100.0
5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	230.0	230.0	230.0	230.0	227.0	229.5	230.0	228.9	226.3	230.0	230.0	230.0
8	120.0	120.0	120.0	120.0	116.7	119.6	120.0	118.9	116.3	120.0	120.0	120.0
9	120.0	120.0	120.0	120.0	117.0	119.4	120.0	118.8	116.3	120.0	120.0	0.0
10	110.0	110.0	110.0	110.0	107.3	109.5	110.0	108.8	106.3	110.0	110.0	110.0
11	170.0	170.0	170.0	170.0	166.9	169.5	170.0	168.8	166.5	170.0	170.0	170.0
12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0
14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Total	**1513**	**1520**	**1544**	**1538**	**1472**	**1487**	**1546**	**1483**	**1468**	**1548**	**1531**	**1430**

6.2 Programación de Predespacho mediante algoritmo Branch and Bound

El programa de predespacho mediante el algoritmo Branch and Bound se realizó utilizando el programa Matlab versión 6.0. Este programa consta de las siguientes rutinas principales.

function Branch_and_Bound()

Programa principal que recibe como datos de entrada los parámetros técnicos de las unidades, los requerimientos de demanda y reserva, las funciones de costo de las unidades y las condiciones iniciales del problema. Entrega como resultado, la matriz de variables de decisión (ON/OFF), las potencias asociadas a las unidades y el costo total del predespacho.

El programa trabaja la optimización sobre un vector de variables denominado X, el cual contiene las variables de potencia y las variables binarias de decisión para cada unidad y cada período de la siguiente forma para un predespacho de N unidades y T etapas:

$$X = \begin{bmatrix} P_{11} & \cdots & P_{1T} \\ \vdots & \ddots & \vdots \\ P_{N1} & \cdots & P_{NT} \\ U_{11} & \cdots & U_{1T} \\ \vdots & \ddots & \vdots \\ U_{N1} & \cdots & U_{NT} \end{bmatrix}$$

Donde:

Pjk corresponde a la potencia asignada a la unidad j en la etapa k.
Ujk corresponde a la variable binaria de decisión de despacho (1: ON, 0: OFF) de la unidad j en la etapa k.

function FC1()

Corresponde a la subrutina que calcula el costo de operación de una unidad para una determinada potencia.

function FO()

Corresponde a la función encargada de calcular la función objetivo del problema de predespacho y por lo tanto representará la función objetivo de cada subproblema del árbol de búsqueda.

function mycon2()

Esta rutina define las restricciones no lineales asociadas a cada subproblema del árbol de búsqueda. En este caso entrega sólo restricciones de igualdad, siendo la principal la asociada al cumplimiento del balance energético para cada etapa.

Adicionalmente, recibe como dato de entrada la matriz R que representa las restricciones asociadas a la matriz binaria de decisión que se agregan en cada nodo y las introduce como restricción en la función de optimización ejecutada en cada subproblema.

function factibilidad()

Esta subrutina recibe como entrada los tiempos mínimos de operación y detención de las unidades, las condiciones iniciales y la matriz X resultante de la optimización de un subproblema, y verifica que todas las variables de decisión sean números binarios y el cumplimiento de los límites de potencia de las unidades.

6.2.1 Diagrama de flujo del programa de predespacho con algoritmo Branch and Bound

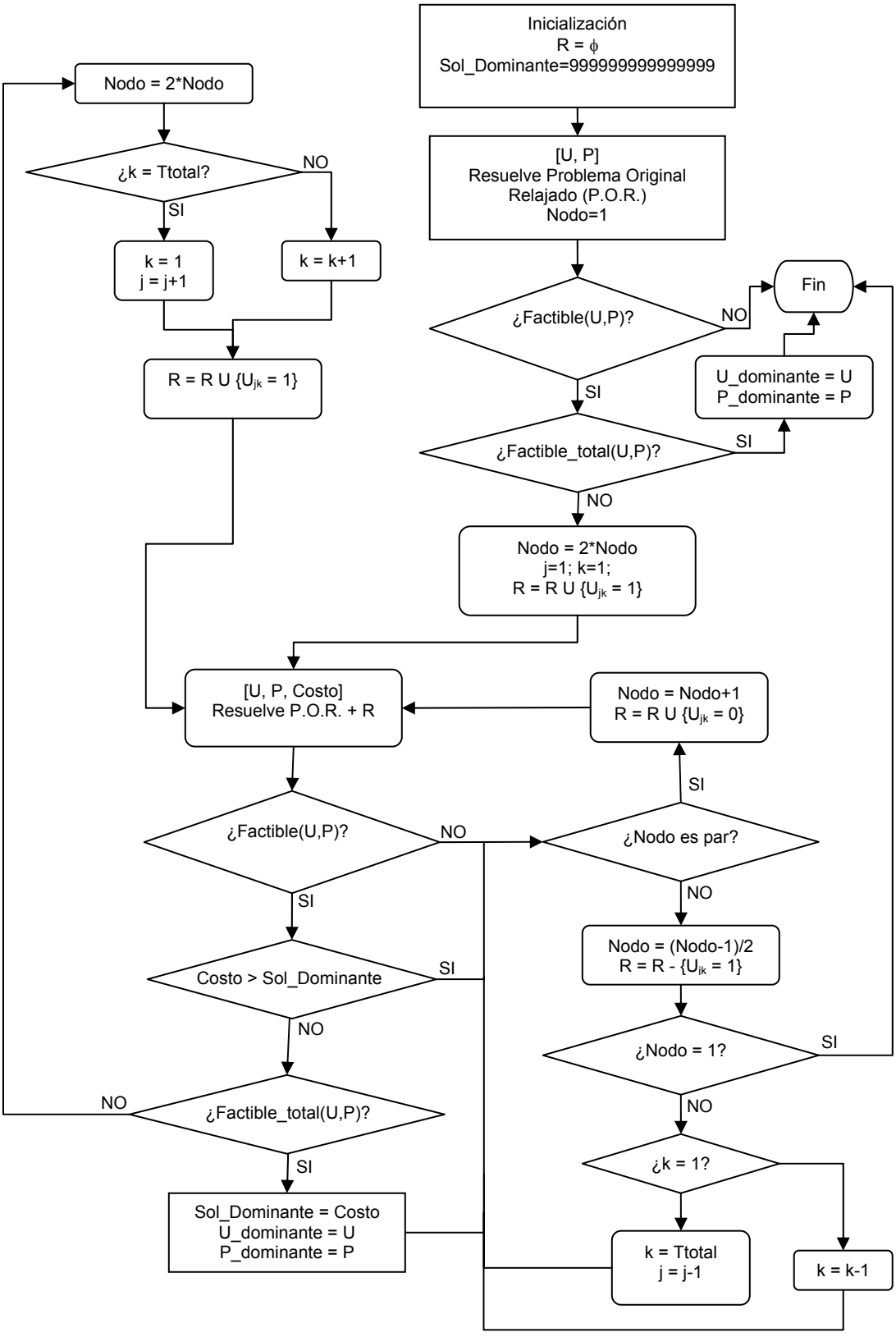
La estructura del programa presenta 3 verificaciones de factibilidad aplicadas a la solución de los subproblemas del árbol de búsqueda, la primera corresponde a verificar si el problema de optimización converge, es decir existe solución. La segunda verificación corresponde a determinar si la restricción asignada a un subproblema es factible respecto de las restricciones agregadas en los subproblemas que lo preceden y el cumplimiento de los tiempos mínimos de operación y detención de las unidades.

Finalmente se tiene una verificación de factibilidad, denominada en el diagrama de flujo como Factible_total, la cual corresponde a verificar la factibilidad de la solución encontrada en un subproblema, para resolver el problema maestro de programación entera, es decir, en ella se verifica que la matriz de decisión obtenida contenga sólo valores binarios.

Sobre la notación utilizada en el diagrama cabe notar lo siguiente:

- R: Corresponde al conjunto de restricciones del tipo $\{U_{jk} = 1\}$.
- Al inicio del algoritmo R = ϕ (conjunto vacío)
- $R \cup \{U_{kj} = 1\}$ denota la unión del conjunto R con el conjunto $\{U_{jk} = 1\}$
- $R - \{U_{kj} = 0\}$ denota la extracción del conjunto $\{U_{jk} = 0\}$ desde el conjunto R.

La estructura del programa se muestra en el siguiente diagrama de flujo:



6.2.2 Prueba preliminar al programa

Se efectuaron pruebas mediante la resolución de problemas de predespacho a pequeños sistemas ficticios, a continuación se muestran los resultados obtenidos en una de estas pruebas.

Datos técnicos de las unidades

Nº Unidad	Pmax MW	Pmin MW	Ton horas	Toff horas
1	550	250	12	3
2	520	250	12	7
3	445	125	10	5
4	150	50	3	2
5	320	120	8	2
6	120	25	3	2
7	280	75	6	6
8	80	20	3	5
9	100	30	2	2
10	60	10	3	2

Ton: corresponde al tiempo mínimo de operación de las unidades.
Toff: corresponde al tiempo mínimo de detención de las unidades.

Condiciones iniciales

Nº Unidad	ton horas	toff horas
1	20	-
2	20	-
3	20	-
4	10	-
5	10	-
6	10	-
7	10	-
8	-	20
9	-	10
10	-	20

ton: corresponde al número de horas en que la unidad estuvo encendida previo al inicio del período de evaluación.

toff: corresponde al numero de horas en que la unidad estuvo apagada previo al inicio del período de evaluación.

Costo de operación

El costo de operación de las unidades se modela mediante funciones lineales que corresponden a considerar unidades con costo variable fijo para cada unidad, luego el costo de operación se obtiene de la ponderación de dicho costo con la potencia generada. A continuación se muestra el costo variable de cada unidad.

Nº Unidad	Cvar $/MW
1	2.053
2	2.058
3	2.059
4	2.313
5	2.342
6	2.422
7	2.523
8	2.545
9	2.605
10	2.759

Datos del predespacho

Se considera un período de 5 etapas horarias y no se imponen requerimientos mínimos de reserva. En la tabla siguiente se muestra la demanda horaria del sistema para cada hora:

hora	Demanda MW
1	2000
2	1980
3	1940
4	1900
5	1840

Resultados

El resultado obtenido es el siguiente:

Costo Total = $ 20.479

Tiempo de ejecución: 4747 segundos (~ 1 hora 19 minutos)

Matriz de decisión:

	Encendido / Apagado				
Unidad/Etapa	**1**	**2**	**3**	**4**	**5**
1	1	1	1	1	1
2	1	1	1	1	1
3	1	1	1	1	1
4	1	1	1	1	1
5	1	1	1	1	1
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	1	0	0	0	0

Potencias Despachadas:

	Generación en MW				
Unidad/Etapa	**1**	**2**	**3**	**4**	**5**
1	550.0	550.0	550.0	550.0	550.0
2	520.0	520.0	520.0	520.0	520.0
3	445.0	445.0	445.0	445.0	445.0
4	150.0	150.0	150.0	150.0	150.0
5	320.0	315.0	275.0	235.0	175.0
6	115.6	119.6	115.6	111.7	105.8
7	275.6	279.6	275.6	271.7	265.8
8	75.6	79.6	75.6	71.7	65.8
9	95.6	99.6	95.6	91.7	85.8
10	15.0	49.7	55.5	51.7	45.8

6.3 Aplicación de Programas en Sistema Reducido

Se realizará una primera comparación de los métodos de Relajación Lagrangeana y Branch and Bound programados mediante la resolución de un problema simple conformado por 10 unidades generadoras con costos marginales representados por funciones lineales y para un período de observación de 5 etapas.

Descripción del problema

Datos técnicos de las unidades

Nº Unidad	Pmax MW	Pmin MW	Ton horas	Toff horas
1	550	250	12	3
2	520	250	12	7
3	445	125	10	5
4	150	50	3	2
5	320	120	8	2
6	120	25	3	2
7	280	75	6	6
8	80	20	3	5
9	100	30	2	2
10	60	10	3	2

Ton: corresponde al tiempo mínimo de operación de las unidades.
Toff: corresponde al tiempo mínimo de detención de las unidades.

Condiciones iniciales

Nº Unidad	ton horas	toff horas
1	20	-
2	20	-
3	20	-
4	10	-
5	10	-
6	10	-
7	10	-
8	-	20
9	-	10
10	-	20

- ton: corresponde al número de horas en que la unidad estuvo encendida previo al inicio del período de evaluación.
- toff: corresponde al numero de horas en que la unidad estuvo apagada previo al inicio del período de evaluación.

Costo de operación

El costo de operación de las unidades se modela mediante funciones lineales. A continuación se muestra la función de costo FC(P) asociada a cada unidad.

Nº Unidad	FC(P) $
1	500+2.053*P
2	250+2.058*P
3	360+2.059*P
4	280+2.313*P
5	650+2.342*P

Nº Unidad	FC(P) $
6	800+2.422*P
7	260+2.523*P
8	500+2.545*P
9	700+2.605*P
10	368+2.759*P

Datos del predespacho

Se considera un período de 5 etapas horarias y no se imponen requerimientos mínimos de reserva. En la tabla siguiente se muestra la demanda horaria del sistema para cada hora:

hora	Demanda MW
1	2000
2	1980
3	1940
4	1900
5	1840

Resolución con algoritmo Relajación Lagrangeana

Para determinar el valor del parámetro h de longitud de paso que entrega la mejor solución para el problema planteado se efectuó el siguiente procedimiento experimental.

Se resolvió el problema considerando la realización de 100 iteraciones, para valores de h enteros desde 1 a 10. En la siguiente tabla se muestran las mejores soluciones obtenidas para cada valor de h.

h	Primal [$]	Brecha Dual %	Nº Iteración
1	30174	9.1	3
2	30174	6.1	7
3	30174	9.1	1
4	30174	5.7	4
5	30174	5.0	7
6	30174	5.2	72
7	30788	6.7	59
8	30174	4.9	47
9	30788	8.1	54
10	30174	5.8	5

Exceptuando los resultados obtenidos para h=7 y h=9, para el resto se obtuvo la misma solución y sólo se difiere en la brecha dual y el número de iteraciones en que esta solución es encontrada.

Si para seleccionar la alternativa de mejor rendimiento consideramos tanto el tiempo de ejecución (número de iteraciones) y la calidad de la solución dual (menor Brecha dual), la mejor alternativa corresponde a h=5.

A continuación se detallará la solución encontrada para una longitud de paso h=5.

Matriz de decisión:

	Encendido / Apagado				
Unidad/Etapa	**1**	**2**	**3**	**4**	**5**
1	1	1	1	1	1
2	1	1	1	1	1
3	1	1	1	1	1
4	1	1	1	1	1
5	1	1	0	0	0
6	0	0	0	0	0
7	1	1	1	1	1
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Potencias Despachadas:

	Generación en MW				
Unidad/Etapa	**1**	**2**	**3**	**4**	**5**
1	550	550	550	550	550
2	520	520	520	520	520
3	445	445	445	445	445
4	150	150	150	150	150
5	260	240	0	0	0
6	0	0	0	0	0
7	75	75	275	235	175
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Costo Total = $ 30174
Tiempo de ejecución = 27.6 segundos

Resolución con algoritmo Branch and Bound

Para el problema descrito anteriormente la solución óptima encontrada por el algoritmo Branch and Bound es la siguiente:

Matriz de decisión:

	Encendido / Apagado				
Unidad/Etapa	**1**	**2**	**3**	**4**	**5**
1	1	1	1	1	1
2	1	1	1	1	1
3	1	1	1	1	1
4	1	1	1	1	1
5	0	0	0	0	0
6	0	0	0	0	0
7	1	1	1	1	1
8	0	0	0	0	0
9	0	0	0	0	0
10	1	1	0	0	0

Potencias Despachadas:

	Generación en MW				
Unidad/Etapa	**1**	**2**	**3**	**4**	**5**
1	550	550	550	550	550
2	520	520	520	520	520
3	445	445	445	445	445
4	150	150	150	150	150
5	0	0	0	0	0
6	0	0	0	0	0
7	280	280	275	235	175
8	0	0	0	0	0
9	0	0	0	0	0
10	55	35	0	0	0

Costo Total = $ 29722
Tiempo de ejecución = 1399.7 segundos

Claramente el tiempo de ejecución obtenido con este método es mucho mayor que el requerido por el método Relajación Lagrangeana, debido principalmente a la complejidad de los problemas de optimización que se deben resolver en cada método y a la cantidad de operaciones matemáticas requeridas por el algoritmo Branch and Bound para encontrar la solución.

El algoritmo Branch and Bound entrega la solución óptima del problema a diferencia del algoritmo Relajación Lagrangeana que entrega una solución en torno al óptimo.

El método de Branch and Bound se puede acelerar considerando una búsqueda también en torno al óptimo, para ello lo que se hace es modificar el criterio de poda en el árbol de búsqueda, eliminando las ramas cuyas soluciones parciales presenten un valor mayor o igual a la solución dominante más un determinado valor ξ.

La diferencia entre las soluciones encontradas por el método Relajación Lagrangeana y el método Branch and Bound es de $452, por lo tanto se ejecutará el método de Branch and Bound considerando $\xi = 452$. Con esto se asegura obtener una solución mejor o igual a la obtenida por el método de Relajación Lagrangeana.

Los resultados obtenidos con esta nueva condición son los siguientes:

Matriz de decisión:

	Encendido / Apagado				
Unidad/Etapa	**1**	**2**	**3**	**4**	**5**
1	1	1	1	1	1
2	1	1	1	1	1
3	1	1	1	1	1
4	1	1	1	1	1
5	0	0	0	0	0
6	0	0	0	0	0
7	1	1	1	1	1
8	1	1	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Potencias Despachadas:

	Generación en MW				
Unidad/Etapa	**1**	**2**	**3**	**4**	**5**
1	550	550	550	550	550
2	520	520	520	520	520
3	445	445	445	445	445
4	150	150	150	150	150
5	0	0	0	0	0
6	0	0	0	0	0
7	280	280	275	235	175
8	55	35	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0

Costo Total = $ 29967
Tiempo de ejecución = 692.3 segundos

Se observa que la solución encontrada tiene un costo un 0.8 % mayor a la solución óptima, sin embrago, el tiempo de ejecución se redujo en un 50%. Por otra parte, el algoritmo de Relajación Lagrangeana entregó una solución un 1.5 % mayor a la óptima pero el tiempo de ejecución corresponde a un 2% del tiempo que demora el método de Branch and Bound en encontrar la solución óptima.

7 APLICACIÓN DE ALGORITMOS PARA EL PREDESPACHO EN EL SING

7.1 Descripción General del Sistema Interconectado del Norte Grande

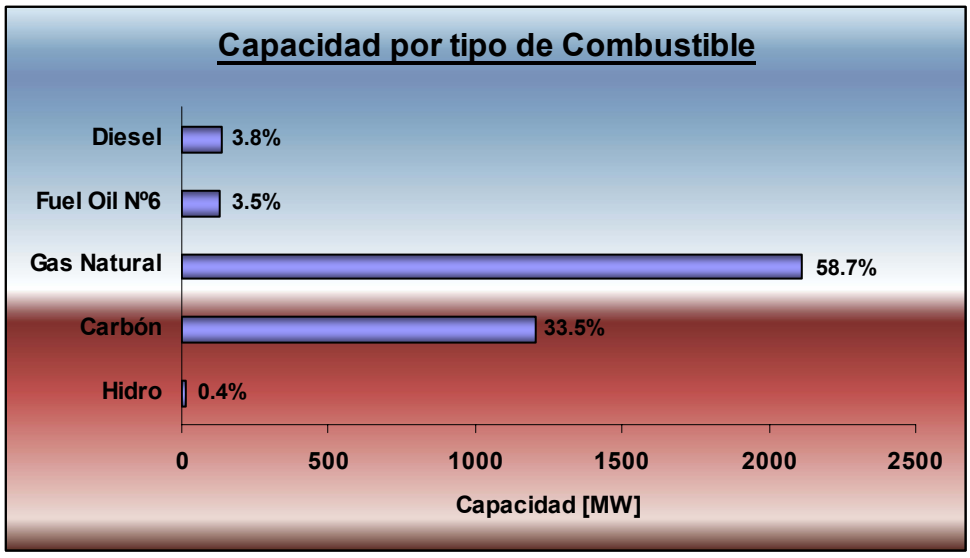
El Sistema Interconectado del Norte Grande (SING) se extiende entre Tarapacá y Antofagasta, Primera y Segunda regiones de Chile, respectivamente, cubriendo una superficie de 185.142 km2, equivalente a 24,5% del territorio continental.

En la actualidad, según cifras del censo de 2002, la población alcanza al 6,1% del total nacional y está concentrada principalmente en algunas ciudades y poblados muy distanciados entre sí. Se pueden identificar las siguientes características importantes del SING:

- Escasos recursos de agua para usos de generación eléctrica.
- Centros de consumo de electricidad separados por grandes distancias.
- Consumo de energía corresponde principalmente a empresas mineras.

El SING actualmente cuenta con una capacidad instalada de 3595.8 MW en 72 unidades generadoras, de las cuales 5 corresponden a unidades de ciclo combinado a gas natural y 9 son del tipo vapor-carbón, entre ambas completan aproximadamente el 92% de la capacidad total del sistema.

La generación bruta máxima horaria en el año 2006 fue de 1770 MW, mientras que el aporte de energía bruta anual acumulado durante el año 2006 fue igual a 13.236 GWh.



7.1.1 Operación Económica

La operación económica del SING privilegia el despacho de las unidades de menor costo variable de producción. Se define el costo variable de producción de una unidad generadora al producto de su consumo específico de combustible por el precio del mismo, más un costo variable no combustible, atribuible fundamentalmente a repuestos, aditivos químicos y lubricantes.

Para poder comparar adecuadamente los costos de generación de cada unidad generadora, se elabora una tabla de costos variables, que contiene el costo variable de producción de cada unidad generadora referido al centro de carga del sistema o nudo básico, mediante el uso de factores que consideran las pérdidas marginales de la red de transmisión (factores de penalización). El centro de carga corresponde actualmente al nudo CRUCERO 220 kV.

La planificación de la operación y el cálculo de los costos marginales se realiza semanalmente, resultando un programa de generación en el cual se considera la previsión horaria de la demanda, los mantenimientos de las unidades generadoras y del sistema de transmisión, así como las limitaciones técnicas de las unidades generadoras, entre las que se cuentan los límites de potencia máxima y mínima, tiempos de puesta en servicio y tiempo mínimo de permanencia en servicio.

El Centro de Despacho y Control del CDEC-SING, coordina en tiempo real con los correspondientes Centros de Control de las empresas integrantes la ejecución del programa diario, realizando en tiempo real las correcciones en la operación, necesarias para absorber las variaciones o desviaciones respecto a lo programado.

Otro aspecto importante de destacar es que las unidades del SING no consideran dentro de su costo de operación costos asociados a la partida ni a la detención de las unidades, sin embargo, los tiempos mínimos de operación y detención son altos en especial para las unidades del tipo vapor-carbón (24 a 120 horas) con lo cual se regula de cierta forma el número de entradas y salidas de unidades en el corto plazo.

En el anexo 7 se muestra un listado con todas las unidades generadoras del SING y sus principales características.

7.1.2 Políticas de Operación

La planificación de la operación de corto plazo en el SING considera un conjunto de políticas de operación tendientes a preservar la seguridad de servicio y la continuidad del suministro, entre ellas destacan las siguientes:

- Reserva en giro: todas las unidades generadoras en operación deben mantener una reserva, la cual corresponde a un porcentaje de su generación máxima, actualmente 7%, la cual puede variar hasta un 5% dependiendo de las necesidades del sistema.
- Reserva en giro primaria: En todo instante de tiempo se debe mantener una cantidad mínima de reserva primaria disponible en el sistema. Dicha cantidad dependerá de la demanda del sistema y de la cantidad de demanda conectada a los relés de desconexión automática de carga. La reserva giro primaria de las unidades corresponde a la reserva de potencia en giro instantánea aportada por las unidades en servicio, en un tiempo menor a 10 segundos, después de ocurrida una contingencia abrupta. Algunas unidades debido a

sus características técnicas deben reducir su generación máxima en una cantidad menor a lo requerido por la reserva en giro para que su aporte de reserva primaria sea efectivo.

- Potencia máxima: algunas unidades de ciclo combinado son despachadas con una potencia máxima inferior debido a las restricciones impuestas por el criterio n-1. Esta potencia máxima dependerá principalmente de la cantidad de demanda conectada a los relés de desconexión automática de carga y la reserva en giro primaria disponible.

Del conjunto de unidades generadoras seleccionadas, las que deben reducir su generación para posibilitar el aporte de reserva en giro primaria son las siguientes: CTTAR (40 MW) y CC2 (TG2A+TG2B+TV2C en 327 MW).

7.2 Modelo Reducido del SING

Para determinar el modelo reducido del SING que se utilizará para probar los programas de los algoritmos de predespacho creados, se tendrán en cuenta las siguientes consideraciones:

- Se considerarán las unidades cuyos tiempos mínimos de operación y detención sean mayores a la duración de una hora.

- Debido a que las unidades del tipo Vapor – Carbón y Ciclos Combinados a gas natural y diesel, completan más del 90 % de la capacidad del sistema se considerarán todas las unidades de este tipo.

- Se tendrán presentes las restricciones de suministro de gas natural que actualmente afectan a las unidades del SING, de esta forma se considerarán disponibles a lo más 2 ciclos combinados operando con gas natural y 1 ciclo combinado operando con combustible diesel.

- No se considerarán unidades pequeñas del tipo motor debido a que su participación aporta un porcentaje menor de la generación del sistema y sus tiempos mínimos de operación y detención son del orden de minutos, es decir, menos a la duración de una etapa del período de evaluación y su costo variable presenta un valor fijo para todo su rango de potencia, lo cual hace trivial su despacho económico.

- No se considerarán las unidades del tipo turbotas diesel (unidades TG1, TG2, TGIQ, etc,) debido a que estas unidades presentan muy escasa presencia en la operación pues se mantienen preferentemente como reserva fría del sistema y como respaldo para Black Start.

Dado que los requerimientos de reserva en giro del sistema se satisfacen mediante la disminución del aporte máximo de potencia de todas las unidades en servicio en un porcentaje fijo de su potencia nominal, esta restricción por simplicidad se considerará reduciendo la potencia máxima de cada unidad en los datos de entrada de los programas.

7.2.1 Unidades Seleccionadas

De acuerdo a los criterios descritos en el ítem anterior las unidades seleccionadas para conformar el modelo reducido del SING son las siguientes:

Potencias Nominales de las Unidades

Nº	Unidad	Tipo	Combustible	Potencia Máxima MW	Potencia Mínima MW	Potencia Despacho (1) MW
1	CTM1	Vapor-Carbón	Carbón	165.9	90	154
2	CTM2	Vapor-Carbón	Carbón	175	90	163
3	CTTAR	Vapor-Carbón	Carbón	158	100	140
4	NTO1	Vapor-Carbón	Carbón	136.3	65	127
5	NTO2	Vapor-Carbón	Carbón	141.04	65	131
6	TG11 + 0.5 TV10	Ciclo Combinado	Gas natural	321.4	185	300
7	TG2A + TG2B + TV2C	Ciclo Combinado	Diesel	400	310	327
8	U10	Vapor - Fuel Oil Nº6	Fuel Oil Nº6	37.5	15	35
9	U11	Vapor - Fuel Oil Nº6	Fuel Oil Nº6	37.5	15	35
10	U12 (Carbón)	Vapor-Carbón	Carbón	85.3	50	79
11	U13 (Carbón)	Vapor-Carbón	Carbón	85.5	50	80
12	U14 (Carbón)	Vapor-Carbón	Carbón	128.3	75	119
13	U15 (Carbón)	Vapor-Carbón	Carbón	130.3	75	121
14	U16-TG + U16-TV	Ciclo Combinado	Gas natural	400	202	320

Fuente: www.cdec-sing.cl

(1) Considera las reducciones por reserva en giro, para aportar reserva en giro primaria (unidades 3, 6) y despacho máximo (unidades 7 y 14).

Tiempos Mínimos de Operación y Detención

Nº	Unidad	Tiempo Mínimo Operación horas	Tiempo Mínimo Detención horas
1	CTM1	120	48
2	CTM2	120	48
3	CTTAR	48	48
4	NTO1	48	48
5	NTO2	48	48
6	TG11 + 0.5 TV10	24	4
7	TG2A + TG2B + TV2C	0	2
8	U10	24	8
9	U11	24	8
10	U12 (Carbón)	48	24
11	U13 (Carbón)	48	24
12	U14 (Carbón)	48	24
13	U15 (Carbón)	48	24
14	U16-TG + U16-TV	0	0

Fuente: www.cdec-sing.cl

7.2.2 Modelamiento de Funciones de Costo

Los costos de operación de las unidades en el SING se calculan mediante su costo variable de producción que corresponde al costo de generar 1 MW de potencia. Este costo de producción varía para distintos intervalos de potencia en el rango de operación de la unidad. Por ejemplo, en la siguiente tabla se muestra los costos de producción de las unidades CTM1 y U14 del día 29 de octubre de 2007:

	Rango de Potencia		
	Pmin	Pmax	Costo Var.
Unidad	MW	MW	[mills/kWh]
CTM1	97	134.9	36.95
CTM1	135	152.4	37.44
CTM1	152.5	165.9	37.47
CTM1	90	96.9	38.06
U14	114	128.2	35.65
U14	88.8	113.9	36.16
U14	65	88.7	37.31

Luego, el costo de producción de las unidades corresponderá a la ponderación de la potencia generada por el costo variable en el tramo de potencia correspondiente.

Claramente la función de costo de las unidades corresponde a una función discreta y no convexa, luego se debe modelar una función de costo para la aplicación en los programas que sea convexa y continua, además es deseable por simplicidad que sea lineal.

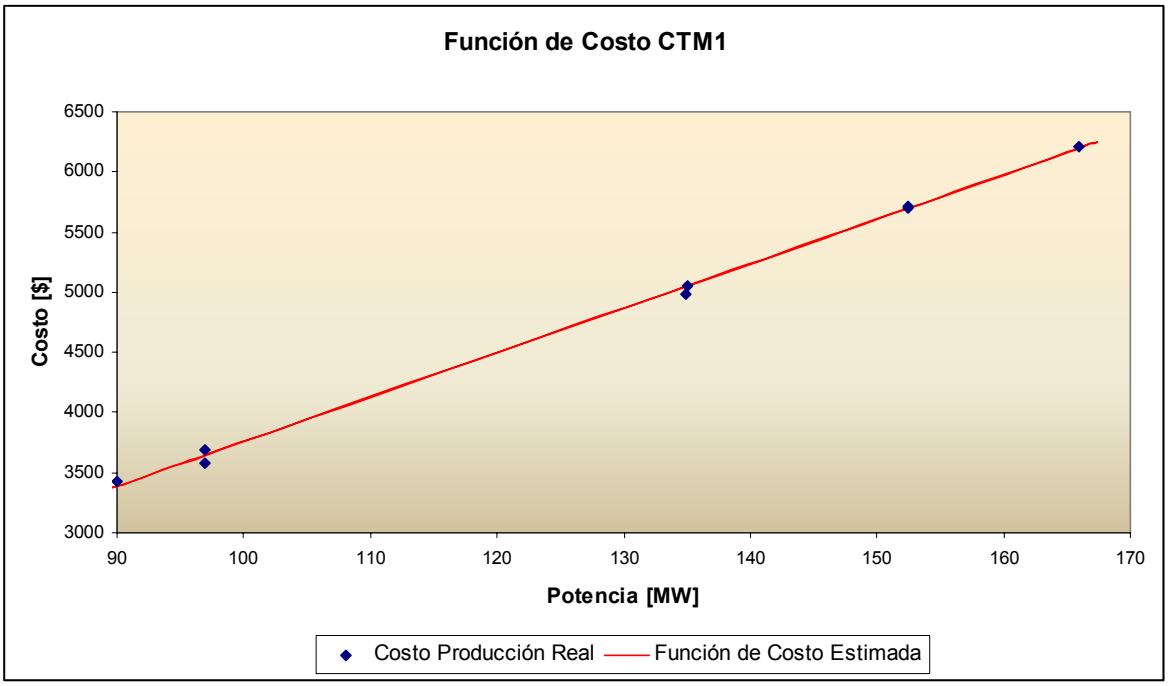
Para obtener dicha función se utilizó la técnica de mínimos cuadrados, de manera de obtener una función lineal que aproxime la función de costo real de las unidades. Para ello se consideraron sólo los puntos extremos de cada tramo de potencia definido en la tabla de costos variable, de esta forma se obtuvieron los coeficientes que constituyen la función de costo lineal de cada unidad. Considerando la tabla de costo variable del SING vigente al 29 de octubre de 2007 se obtienen las funciones de costo estimadas que se muestran en la tabla siguiente.

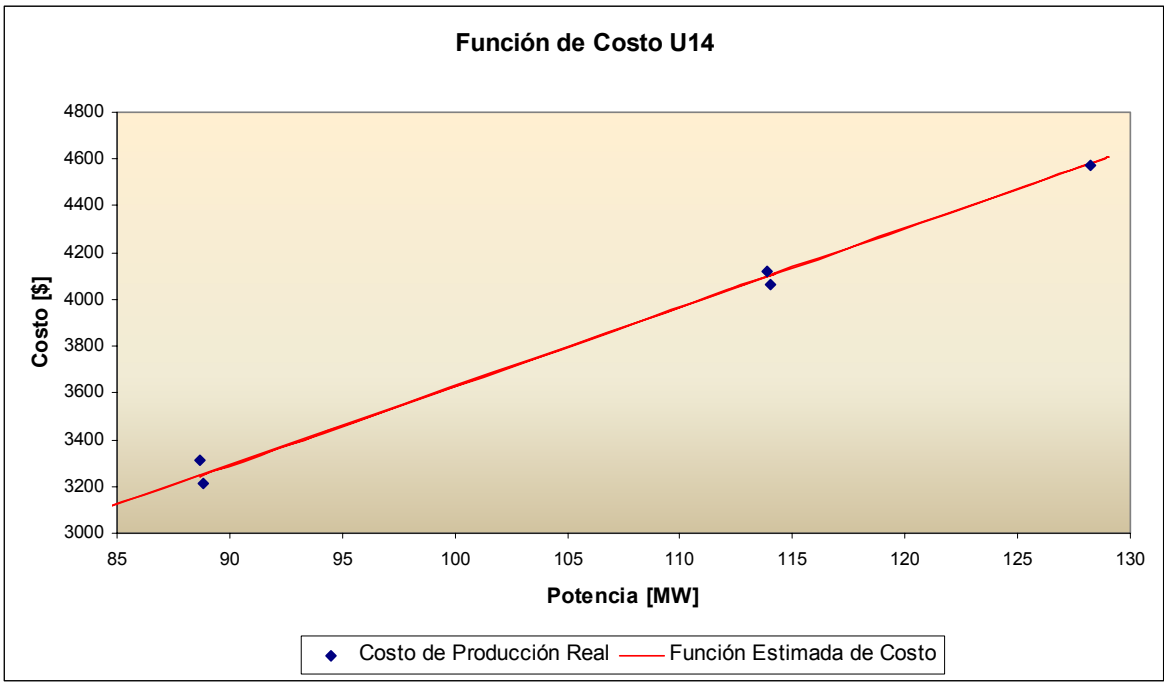
Funciones de costo estimadas

		FC(P)			FC(P)
Nº	Unidad	[$]	Nº	Unidad	[$]
1	CTM1	37 × P + 58	8	U10	118 × P + 574
2	CTM2	36 × P + 107	9	U11	118 × P + 574
3	CTTAR	37 × P + 121	10	U12 (Carbón)	41 × P + 7.8
4	NTO1	33 × P + 204	11	U13 (Carbón)	40 × P - 41
5	NTO2	31 × P + 388	12	U14 (Carbón)	34 × P + 249
6	TG11 + 0.5 TV10	12 × P + 1212	13	U15 (Carbón)	32 × P + 301
7	TG2A + TG2B + TV2C	130 × P + 4041	14	U16-TG + U16-TV	30 × P + 836

En las figuras siguientes se presenta gráficamente la aproximación lineal obtenida para la función de costo de producción de las unidades CTM1 y U14.

Aproximación Lineal de las funciones de costo de producción de CTM1 y U14





7.3 Aplicación de Algoritmos para el Predespacho del SING

Se realizará el cálculo de un predespacho para el modelo reducido del SING en un período de observación de 24 horas dividido en etapas horarias.

Se considerarán las siguientes restricciones:

- Balance Energía (Demanda = Generación).
- Tiempos mínimos de operación y detención de las unidades.
- Capacidad de las unidades (considera porcentaje de reserva en giro)

La demanda utilizada corresponde a la generación bruta real del SING del 23 de agosto de 2007.

Hora	Demanda [MW]	Hora	Demanda [MW]
1	1625.9	13	1541.4
2	1584.5	14	1510.2
3	1559.5	15	1514.8
4	1556.8	16	1532.4
5	1547.4	17	1570.5
6	1534.9	18	1568.6
7	1554.9	19	1618.6
8	1561.9	20	1670.8
9	1533.6	21	1673.8
10	1545.2	22	1703.6
11	1566.3	23	1680.7
12	1559.0	24	1661.2

Aplicación Algoritmo de Branch and Bound

Los resultados de la aplicación del programa Branch and Bound son los siguientes:

- Tiempo de Ejecución: 6,3 horas
- Costo total: $ 1.200.757

Aplicación Algoritmo de Relajación Lagrangeana

Los resultados de la aplicación del programa de Relajación Lagrangeana son los siguientes:

- Tiempo de Ejecución: 23 minutos
- Costo total: $ 1.205.247
- Nº de iteraciones: 150
- Brecha dual: 0.5 %

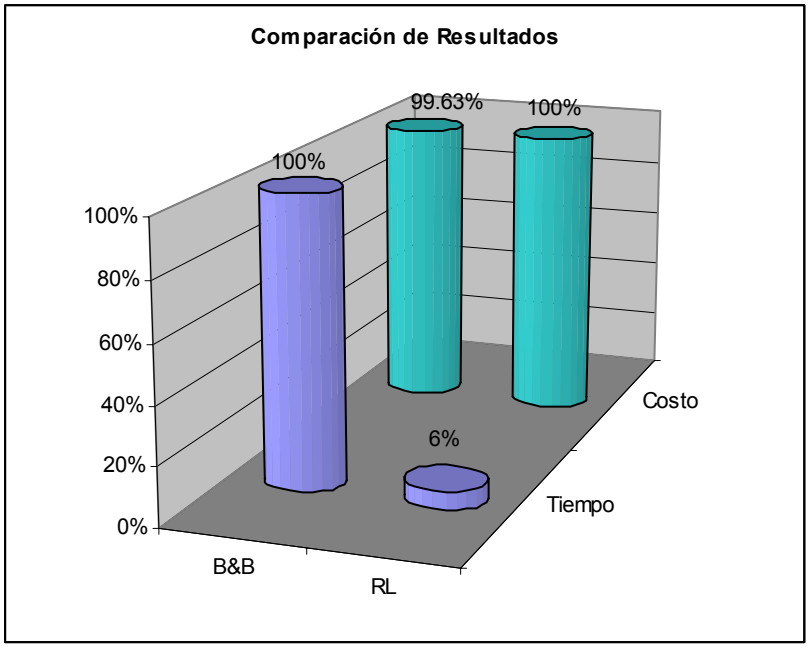
En el anexo 6 se muestran las matrices con las potencias horarias de las unidades despachadas y las matrices binarias de decisión para ambos algoritmos.

7.3.1 Comparación de Resultados

Se observa al igual que las soluciones obtenidas para el sistema reducido en el punto 6.3, el costo total obtenido mediante el algoritmo Branch and Bound presenta un valor inferior a la solución encontrada mediante la técnica de Relajación Lagrangeana. Para este caso, la mejora alcanza un 0.4% (valor similar a la brecha dual).

Por otra parte, el tiempo de operación requerido por el método de Relajación Lagrangeana sólo alcanza a un 6% respecto del tiempo requerido por el otro algoritmo, sin embargo, se debe considerar que el algoritmo de Relajación Lagrangeana requirió un tiempo adicional para sintonizar el parámetro de longitud de paso h, para conseguir soluciones factibles. En todo caso el tiempo empleado en la sintonización de este parámetro es muy inferior al tiempo requerido en la ejecución del algoritmo.

En el gráfico siguiente se ilustran en forma comparativa los rendimientos obtenidos por cada algoritmo.



8 CONCLUSIONES.

Hacer una comparación técnica del desempeño de dos algoritmos de optimización combinatorial, es una tarea muy compleja, dado que intervienen un gran número de factores que definen la calidad del desempeño, entre ellos destacan: la calidad de las soluciones encontradas (cercanía al óptimo), los tiempos de ejecución de requeridos, la capacidad del algoritmo para encontrar soluciones factibles y el uso de heurística en la búsqueda de soluciones factibles. Otro aspecto importante que influye en el desempeño de estos algoritmos corresponde a las dimensiones y complejidad del problema que se desea resolver.

Como primera impresión se puede concluir que la complejidad matemática que presenta el algoritmo de Relajación Lagrangeana es mucho menor respecto del método de Branch and Bound. Esto fundamentalmente por dos aspectos: la complejidad y dimensiones de los subproblemas de optimización que se deben resolver en cada algoritmo y en segundo lugar por el número de optimizaciones requeridas.

Claramente el algoritmo de Relajación Lagrangeana al resolver subproblemas desacoplados (programación dinámica individual) requiere optimizar funciones de T variables, donde T corresponde al número de etapas del período de evaluación, y posteriormente realizar un despacho económico para cada etapa. En resumen para un sistema de N unidades generadores, en cada iteración este algoritmo requiere resolver N subproblemas de optimización con T variables de tipo entero.

Por otra parte, el algoritmo Branch and Bound resuelve para cada nodo del árbol de búsqueda subproblemas de optimización de N×T variables continuas, y el número de subproblemas que se deben resolver para encontrar el óptimo crece exponencialmente con el tamaño del problema, con un máximo de 2^{N+T}-1 problemas.

Es evidente que la complejidad matemática tiene una relación directa con el tiempo de ejecución de las rutinas, esto en conjunto con la naturaleza enumerativa del árbol de búsqueda del algoritmo Branch and Bound hacen que éste presente, para sistemas de dimensiones reales, un rendimiento en cuanto al tiempo de ejecución requerido, muy inferior al de la Relajación Lagrangeana.

Respecto a la calidad de las soluciones, el algoritmo Branch and Bound entrega la solución óptima del problema, aunque también existe la alternativa de obtener una aproximación al óptimo. Por su parte el algoritmo de Relajación Lagrangeana en general entrega soluciones cercanas al óptimo y la calidad de las soluciones tiende a mejorar en la medida que el sistema evaluado aumenta su dimensión. Se observa que la brecha dual puede variar dependiendo de las dimensiones del problema, el número de iteraciones consideradas y la longitud de paso utilizada en el método del subgradiente, entre un 0.5% y un 5 %.

Una de las principales debilidades del método de Relajación Lagrangeana corresponde a la utilización de heurística, la cual es requerida para inicializar los multiplicadores de Lagrange y podría ser necesaria para generar soluciones factibles, esto hace que el resultado obtenido dependa de la naturaleza de la heurística empleada. Por el contrario, el método Branch and Bound no utiliza heurística para determinar soluciones, lo cual es deseable, sobretodo en ambiente de mercados altamente competitivos.

Otro aspecto importante que se ve afectado por la utilización de heurística corresponde a la consideración en el problema de predespacho de restricciones de mayor complejidad, como por ejemplo, restricciones de transmisión. Mientras mayor sea la complejidad del problema, mayor será la dificultad para encontrar soluciones factibles basadas en heurística, lo cual podría afectar la convergencia de la Relajación Lagrangeana. Para el caso de Branch and Bound el agregar otras restricciones más complejas es relativamente sencillo y no requiere modificar el algoritmo.

En la aplicación práctica de los algoritmos programados, ya sea en la resolución de problemas simples, de pocas unidades y pocas etapas o en problemas de mayor dimensión (modelo simplificado del SING), se observa que el rendimiento del algoritmo Relajación Lagrangeana es muy superior al de el algoritmo Branch and Bound. El tiempo de ejecución requerido por el algoritmo de Relajación Lagrangeana para obtener una solución con una precisión del orden de 99 % o superior, es muy inferior al tiempo empleado por el algoritmo Branch and Bound.

Teniendo en cuenta que las políticas de operación utilizadas en la actualidad en el SING no representan gran complejidad de programación y considerando que, de acuerdo a lo observado no se requiere un uso significativo de heurística para determinar soluciones factibles, se puede inferir que el algoritmo de Relajación Lagranjeana presenta características que lo hacen aplicable para la planificación de la operación de corto plazo del SING.

9 BIBLIOGRAFÍA

[1] Allen J. Wood & Bruce F. Wollenberg, "*Power Generation, Operation and Control*". John Wiley and Sons, Second Edition, NY 1996.

[2] Z. Ouyang, S. M. Shahidehpour, "*An Intelligent Dynamic Programming for Unit Commitment Applications*". IEEE Transactions on Power Systems, Vol. 6, Nº3, Agosto 1991: 1203-1209.

[3] Tomonobu Senjyu, Kai Shimabukuro, Katsumi Uezato, Toshihisa Funabashi, *"A Fast Technique for Unit Commitment Problem by Extended Priority List"*. IEEE Transactions on Power Systems, Vol. 18, Nº 2, Mayo 2003.

[4] Vladimir Tsurkiov, "*Large-Scale - Problems and Methods*", Applied Optimization series. Luwer Academic Publishers. Sept, 2003.

[5] Sudhir Virmani, Eugene C. Adrian, Karl Imhof, Shishir Hukherjee, *"Implementation of a Lagrangian Relaxation Based Unit Commitment Problem"*. IEEE Transactions on Power Systems, Vol. 4, Nº 4, Octubre 1989.

[6] Papadimitriou Christos H., Steiglitz Kenneth*, "Combinatorial Optimization: Algorithms and Complexity"*. Dover Publications Inc. NY, 1998.

[7] Chao-an Li, Raymond B. Johnson, Alva J. Svoboda, *"A New Unit Commitment Method"*. IEEE Transaction on Power System, Vol. 12, Nº 1, Febrero 1997, 113 – 119.

[8] S. A. Kazarlis, A. G. Bakirtzis, V. Petridis, "*A genetic Algorithm Solution to the Unit Commitment Problem*", IEEE Transactions on Power Systems, Vol. 11, Nº 1, Febrero 1996.

[9] George L. Nemhauser, Laurence A. Wolsey, *"Integer and Combinatorial Optimization"*.

[10] William J. Cook, William H. Cunningham, William R. Pulleyblank, Alexander Schrijver, *"Combinatorial Optimization"*.

[11] Xiaohong Guan, Qiaozhu Zhai, Alex Papalexopoulos, *"Optimization Based Methods for Unit Commitment: Lagrangian Relaxation versus General Mixed Integer Programming"*. Power Engineering Society General Meeting, 2003, IEEE.

[12] Parámetros técnicos de unidades generadoras, Generación Bruta de unidades y Tabla de costos variables, www.cdec-sing.cl.

[13] Stuart E. Dreyfus & Averill M Law, "*The Art and Theory of Dynamic Programming*", New York, USA, Academic Press, 1977.

[14] G. B. Sheblé & G. N. Fahd, "*Unit Commitment Literature Synopsis*", IEEE Transactions on Power Systems, Vol. 9, No 1, pp. 128-135, Feb 1994.

[15] Benders J. F., "Partitioning Procedures for Solving Mixed Variables Programming Problems", Numer. Math, 1977.

A. ANEXO 1: DEFINICIONES

Definición 1:

Dos problemas de programación matemática son llamados equivalentes con respecto a la funcional si el valor óptimo de la función objetivo de ambos problemas son idénticos.

Definición 2:

Dos problemas A y B son llamados equivalentes respecto a la variable x si $X_A = X_B$.
Donde:

$$X_A = \{x \mid \exists\, y : (x, y) - \textit{es una solución óptima del problema } A\}$$

$$X_B = \{x \mid \exists\, z : (x, z) - \textit{es una solución óptima del problema } B\}$$

Es decir, X_A y X_B son los conjuntos de la componente x de las soluciones óptimas de los problemas A y B respectivamente.

B. ANEXO 2: DEDUCCIÓN DE LA EXPRESIÓN PARA INICIALIZACIÓN DE MULTIPLICADOR ASOCIADO A LA RESERVA

Supongamos que para un instante de tiempo determinado se requiere para satisfacer la demanda el despacho de N unidades en las potencias $P_1,\ldots,P_N$, luego la reserva en giro disponible en el sistema será la siguiente:

$$R_G = \sum_{i=1}^{N}\left(P_i^{\max} - P_i\right)$$

y el costo de operación del sistema corresponderá a Cop.

Ahora supongamos que se desea aumentar en 1 kWh la reserva en giro del sistema, para ello es necesario el despacho de una unidad adicional, la cual generará una potencia ΔP_{N+1}. La nueva reserva en giro será la siguiente:

$$R'_G = R_G + P_{N+1}^{\max}$$

Luego el aumento de reserva corresponderá a $R'_G - R_G = P_{N+1}^{\max}$.

Por otra parte, si el costo de generar un kWh adicional en el sistema es λ^t entonces el costo de operación asociado al aumento marginal de la reserva será:

$$Cop' = Cop - \lambda^t \cdot \Delta P_{N+1} + \beta_{N+1} \cdot \Delta P_{N+1}$$

donde β_{N+1} corresponde al costo marginal de la unidad *N+1* que fue necesaria para aumentar la reserva del sistema.

El aumento en el costo de operación del sistema corresponderá a lo siguiente:

$$Cop' - Cop = \left(\beta_{N+1} - \lambda^t\right)\cdot \Delta P_{N+1}$$

Luego el aumento de costo asociado al aumento de un kWh de reserva quedará dado por la siguiente expresión:

$$\frac{Cop' - Cop}{R'_G - R_G} = \left(\beta_{N+1} - \lambda^t\right)\cdot \frac{\Delta P_{N+1}}{P_{N+1}^{\max}}$$

La expresión (11) se obtiene considerando los limites operativos de la unidad con lo cual, se debe cumplir que:

$$\Delta P_{N+1} = P_{N+1}^{\min}$$

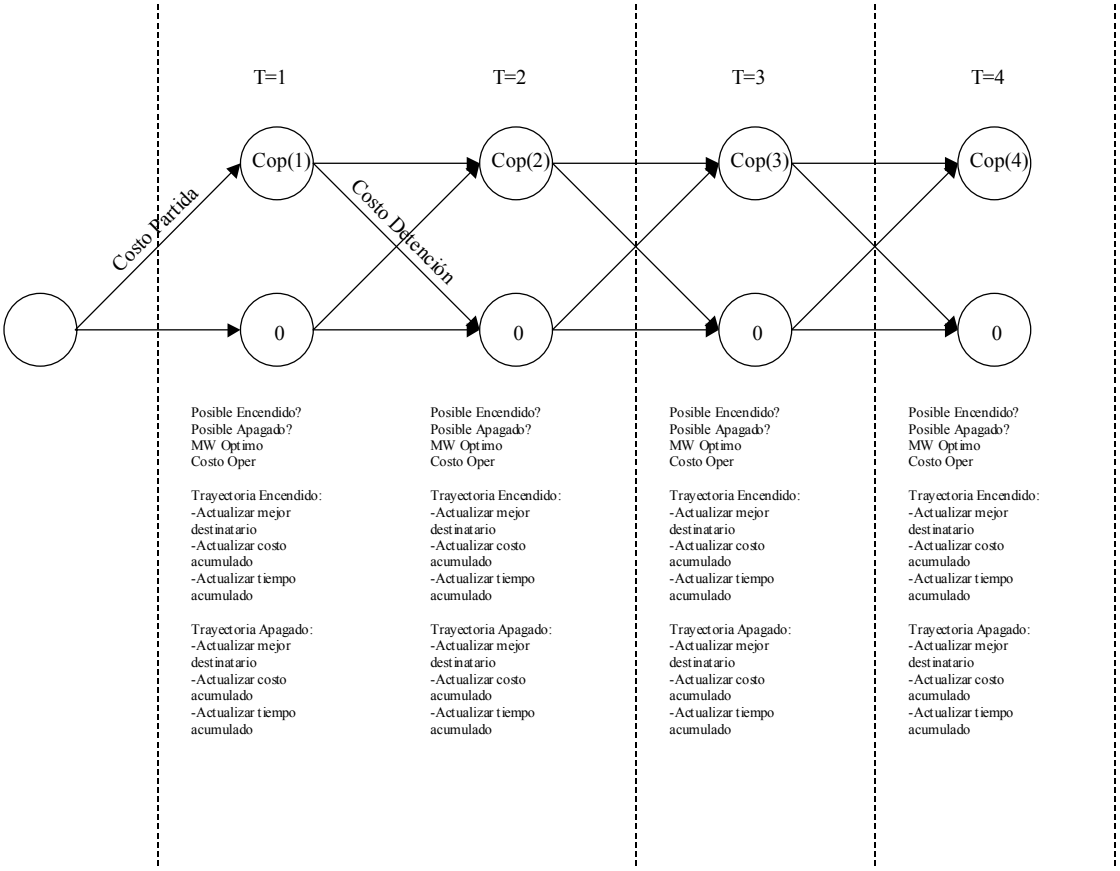
C. ANEXO 3: ALGORITMO DE PROGRAMACIÓN DINÁMICA INDIVIDUAL DE LAS UNIDADES

El problema de optimización a resolver corresponde al siguiente:

$$Q_n = \min \sum_{t=1}^{T} \left[u_{n,t} \cdot FC(P_{n,t}) + S_{n,t}^{start} + S_{n,t}^{stop} - \left(\lambda^t \cdot P_{n,t} \cdot u_{n,t} + \mu_t \cdot (P_{n,t} - P_{n,t}^{\max}) \cdot u_{n,t} \right) \right]$$

esta minimización es con respecto a las potencias de la unidad *n* para todo el período, es decir, $P_{n,t} \ \forall t = 1,...,T$ y las restricciones que se deben considerar corresponden a las restricciones de capacidad (4), reserva en giro de cada unidad (5) y temporales (6).

La presencia de variables binarias de decisión en los costos de operación y los costos de partida conforman un problema de optimización no-convexo entero-mixto. Se plantea encontrar la asignación óptima de la unidad en los períodos *t=1,...,T* teniendo para cada *t*, dos (2) posibles estados de operación: encendido (ON) y apagado (OFF). La figura siguiente muestra el esquema de programación dinámica individual tradicional.



Esquema del algoritmo de Programación Dinámica [1].

El algoritmo de resolución de este problema de optimización tiene las siguientes etapas:

1. Determinación de las condiciones iniciales (*t=0*): se deben considerar las siguientes condiciones iniciales:

   - Estado inicial de la unidad ON: en servicio; OFF: Fuera de servicio.

   - Tiempo acumulado en estado ON; Tiempo acumulado en estado OFF según corresponda.

2. Para *t=1* determinar estados posibles en función de la condición inicial, tiempos mínimos de operación/detención, salidas programadas, etc.

3. Resolver el problema de optimización Q_n para *t=1*:
   - Para el estado OFF, el valor de la función Qn será nulo.
   - En el estado ON, para resolver la función Q_n se debe resolver:

$$\frac{\partial Q_n}{\partial P_{n,t}} = 0 \quad \Rightarrow \quad \frac{\partial}{\partial P_{n,t}}\left[FC_n\left(P_{n,t}\right) - P_{n,t}\cdot\left(\lambda^t + \mu_t\right)\right] = \frac{\partial FC_n\left(P_{n,t}\right)}{\partial P_{n,t}} - \lambda^t - \mu_t = 0$$

   La solución a esta ecuación es $\dfrac{\partial FC\left(P_{n,t}^{opt}\right)}{\partial P_{n,t}} = \lambda^t + \mu_t$.

   Tres casos deben ser considerados dependiendo de la relación entre $P_{n,t}^{opt}$ y los limites operativos de la unidad para evaluar la función objetivo (Q_n) en *t=1*:

   1. Si $P_{n,t}^{opt} \le P_n^{\min}$, entonces:

$$\min\left\{FC_n\left(P_{n,t}\right) - P_{n,t}\cdot\left(\lambda^t + \mu_t\right)\right\} = FC_n\left(P_n^{\min}\right) - P_n^{\min}\cdot\left(\lambda^t + \mu_t\right)$$

   2. Si $P_n^{\min} \le P_{n,t}^{opt} \le P_n^{\max}$, entonces:

$$\min\left\{FC_n\left(P_{n,t}\right) - P_{n,t}\cdot\left(\lambda^t + \mu_t\right)\right\} = FC_n\left(P_{n,t}^{opt}\right) - P_{n,t}^{opt}\cdot\left(\lambda^t + \mu_t\right)$$

   3. Si $P_{n,t}^{opt} \ge P_n^{\max}$, entonces:

$$\min\left\{FC_n\left(P_{n,t}\right) - P_{n,t}\cdot\left(\lambda^t + \mu_t\right)\right\} = FC_n\left(P_n^{\max}\right) - P_n^{\max}\cdot\left(\lambda^t + \mu_t\right)$$

4. Para t=1, cuando el estado inicial (*t=0*) es apagado (OFF), se considerará la posibilidad de encendido de la unidad sólo si $\left\{FC_n\left(P_{n,t}\right) - P_{n,t}\cdot\left(\lambda^t + \mu_t\right)\right\} < 0$, dado que el costo de mantener apagada la unidad es nulo y su encendido tiene asociado un costo de partida.

5. Determinación de la mejor trayectoria para llegar al estado de encendido:

   Si definimos T_t^{OFF} como el costo asociado a la mejor trayectoria para llegar al estado apagado en el tiempo *t* y T_t^{ON} el costo asociado a la mejor trayectoria para llegar al estado de encendido en el tiempo *t*, entonces:

   $Si\ T_{t-1}^{OFF} + S_{n,t}^{start} < T_{t-1}^{ON} \quad \Rightarrow \quad registrar\ como\ mejor\ trayectoria\ para\ llegar\ al\ estado\ encendido$, "*venir desde apagado*".

   $Si\ T_{t-1}^{OFF} + S_{n,t}^{start} > T_{t-1}^{ON} \quad \Rightarrow \quad registrar\ como\ mejor\ trayectoria\ para\ llegar\ al\ estado\ encendido$, "*venir desde encendido*".

6. Determinación de la mejor trayectoria para llegar al estado de apagado:

   Utilizando la misma notación del punto anterior;

   $Si\ T_{t-1}^{ON} + S_{n,t}^{stop} < T_{t-1}^{OFF} \quad \Rightarrow \quad registrar\ como\ mejor\ trayectoria\ para\ llegar\ al\ estado\ apagado$, "*venir desde encendido*".

   $Si\ T_{t-1}^{ON} + S_{n,t}^{stop} > T_{t-1}^{OFF} \quad \Rightarrow \quad registrar\ como\ mejor\ trayectoria\ para\ llegar\ al\ estado\ apagado$, "*venir desde apagado*".

7. Actualizar el costo y el tiempo de operación acumulado en las dos (2) trayectorias (encendido y apagado).

8. Si t = T , entonces, si el costo acumulado hacia estado encendido es menor que el costo acumulado hacia estado apagado, se selecciona la trayectoria hacia el estado encendido para el despacho de la unidad, de lo contrario se selecciona la trayectoria hacia el estado apagado.

9. Si t < T volver a 3.

D. ANEXO 4: MÉTODO DEL SUBGRADIENTE

El método del subgradiente es un algoritmo sencillo para resolver problemas de optimización de funciones convexas no-diferenciables.

Supongamos una función convexa $f : R^N \to R$, para maximizar *f* el método del subgradiente utiliza una búsqueda iterativa de la solución de la siguiente forma:

$$x^{(k+1)} = x^{(k)} + \alpha_k \cdot g^{(k)}$$

donde $x^{(k)}$ corresponde a la *k*-ésima iteración, g^k es cualquier subgradiente de *f* para $x^{(k)}$ y $\alpha_k > 0$ es a longitud de paso para la iteración *k*. De esta forma, para cada iteración del método del subgradiente, se toma un paso en la dirección de un subgradiente positivo.

Se llama subgradiente de *f* en *x* a cualquier vector *g* que satisface la inecuación: $f(y) \geq f(x) + g^T(y - x)$ para todo *y*. Cuando *f* es diferenciable la única elección para $g^{(k)}$ sería $\nabla f\left(x^{(k)}\right)$, convirtiéndose entonces en un método de gradiente convencional con la excepción del control de longitud α_k.

El método del subgradiente no es un método monótonamente creciente, por lo tanto, es usual llevar registro de la mejor solución encontrada hasta entonces, es decir, para cada paso se debe registrar:

$$f_{best}^{(k)} = \max\left\{f_{best}^{(k-1)}, f\left(x^{(k)}\right)\right\}$$

También se debe registrar $i_{best}^{(k)} = k$ si $f\left(x^{(k)}\right) = f_{best}^{(k)}$, es decir si $x^{(k)}$ es el mejor punto encontrado hasta ahora (en métodos descendentes no es necesario llevar tal registro sino que el punto actual siempre corresponderá al mejor hasta el momento).

Reglas de longitud de paso:

Varios tipos de longitud de paso son usados:

a) *Tamaño del paso constante*: $\alpha_k = h$, con *h* una constante independiente de *k*.

b) *Longitud del paso constante*: $\alpha_k = h / \left\|g^{(k)}\right\|_2$, esto se refiere a mantener la norma eucleniana entre soluciones consecutivas constantes en cada iteración, es decir: $\left\|x^{(k+1)} - x^{(k)}\right\|_2 = h$.

c) *Series no sumables pero de cuadrados sumables*: α_k satisface la condición:

$$\sum_{k=1}^{\infty} \alpha_k^2 < \infty, \ \sum_{k=1}^{\infty} \alpha_k = \infty$$

un ejemplo típico de lo anterior es: $\alpha_k = a/(b + k)$, donde $a > 0$ y $b \geq 0$.

d) *Series no sumables diminutivas*: α_k satisface la condición:

$$\lim_{k\to\infty} \alpha_k = 0, \quad \sum_{k=1}^{\infty} \alpha_k = \infty$$

Las longitudes de paso que satisfacen esta condición son llamadas *diminishing step size rule*. Un ejemplo típico corresponde a: $\alpha_k = a/\sqrt{k}$ con $a > 0$

Convergencia:

Para las reglas de longitud de paso a) y b), el algoritmo de subgradiente garantiza la convergencia dentro de un rango en torno al óptimo, es decir se cumple:

$$\lim_{k\to\infty} f_{best}^{(k)} - f^* < \xi$$

donde f^* corresponde al valor óptimo del problema (lo anterior implica que el método del subgradiente permite encontrar un punto sub-óptimo (ξ) en un número finito de pasos). El valor ξ es función del parámetro *h* de tamaño de paso y decrece con este.

Para las reglas c) y d) el algoritmo garantiza la convergencia, es decir:

$$\lim_{k\to\infty} f\left(x^{(k)}\right) = f^*$$

Cuando la función *f* es diferenciable se puede decir que para la regla a), el método del subgradiente converge al óptimo cuando el parámetro *h* es lo suficientemente pequeño.

Prueba de Convergencia

La prueba de convergencia para el método del subgradiente generalmente consiste en medir la distancia euclediana a la solución óptima. Para funciones *f* que satisfacen las condiciones Lipschitz, el empleo del método del subgradiente permite el cumplimiento de la siguiente desigualdad:

$$f_{best}^{(k)} - f^* = \min_{i=1,\ldots,k} : f(x^i) - f^* \leq \frac{R^2 + G^2 \sum_{i=1}^{k} \alpha_i^2}{2\sum_{i=1}^{k} \alpha_i}$$

Donde *G* corresponde a la cota superior de la norma del subgradiente $\|g\|_2$ y *R* se define como una constante proporcional a la norma de la solución inicial. Es decir:

$$R = \left\|x^1 - x^*\right\|_2^2$$

A partir de esta expresión general se pueden deducir las siguientes desigualdades:

- Para *Tamaño del paso constante*: ($\alpha_k = h$)

$$f_{best}^{(k)} - f^* \leq \frac{R^2 + G^2 h^2 k}{2hk}$$

Luego cuando $k \to \infty$ el error del método del subgradiente tiende a $G^2 h / 2$.

- Para *Longitud del paso constante*: ($\alpha_k = h / \left\| g^k \right\|_2$)

$$f_{best}^{k} - f^* \leq \frac{R^2 + G^2 h^2 k}{2hk}$$

Si k $\to \infty$ el error del método del subgradiente se aproximará a $G \cdot h / 2$.

- Para *Series no sumables pero de cuadrados sumables*:

$$f_{best}^{k} - f^* \leq \frac{R^2 + G^2 \left\| \alpha \right\|_2^2}{2\sum_{i=1}^{k} \alpha_i}$$

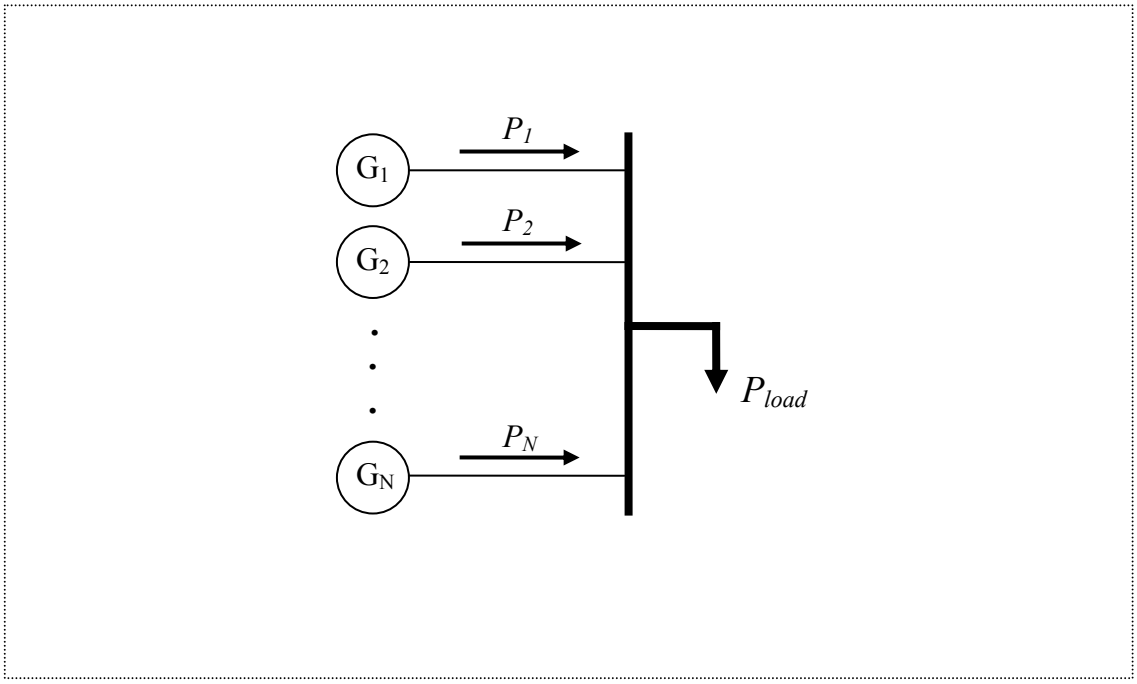
Por lo tanto, si $k \to \infty$ el método del subgradiente converge, es decir, $\lim_{k \to \infty} f\left(x^{(k)}\right) = f^*$:

- Para *Series no sumables diminutivas*:

El método del subgradiente converge cuando $k \to \infty$.

E. ANEXO 5: DESPACHO ECONÓMICO

El despacho económico en un sistema eléctrico se puede representar en forma simplificada como un conjunto de N generadores y una carga conectada a una única barra, como se muestra en la siguiente figura:



Donde G_1, ..., G_N corresponde a los generadores y P_1, ..., P_N las potencias generadas por estos, las cuales deben cumplir con:

$$\sum_{i=1}^{N} P_i = P_{load}$$

Con la representación anterior el problema del despacho económico corresponde a determinar las potencias P_1, ..., P_N que hagan mínimo el costo de operación de este sistema y satisfagan la demanda (P_{load}).

Considerando que FC_k corresponde a la función que representa el costo de operación de la unidada k, la cual depende de la potencia P_k generada, entonces el problema puede ser planteado formalmente de la siguiente forma:

$$\min \sum_{i=1}^{N} FC_i(P_i)$$

Sujeto a las siguientes restricciones:

$$\sum_{i=1}^{N} P_i = P_{load} \;\; ; \; P_i^{\min} \le P_i \le P_i^{\max} \quad \forall\, i = 1,...,N$$

Este problema de optimización se puede resolver mediante diversas técnicas, las más utilizadas son el Método λ-iterativo, el Método de Newton y el Método del Subgradiente. A continuación se

dará una breve explicación de los dos primeros métodos (el método del Subgradiente se encuentra explicado en extenso en el Anexo Nº4) para lo cual es necesario definir la función de lagrange asociada a este problema:

$$L = \sum_{i=1}^{N} FC_i(P_i) + \lambda \cdot \left(\sum_{i=1}^{N} P_i - P_{load} \right) \qquad (1)$$

Método λ-iterativo

Este método se basa en obtener las potencias asociadas a cada unidad a partir del multiplicador λ mediante la resolución de las ecuaciones dadas por la condición de óptimo de la función de lagrange, es decir, que las derivadas parciales de esta función respecto a las potencias deben ser nulas:

$$\frac{\partial L}{\partial P_i} = 0 \quad \Rightarrow \quad \frac{\partial FC_i(P_i)}{\partial P_i} - \lambda = 0 \quad \forall\, i = 1,...,N$$

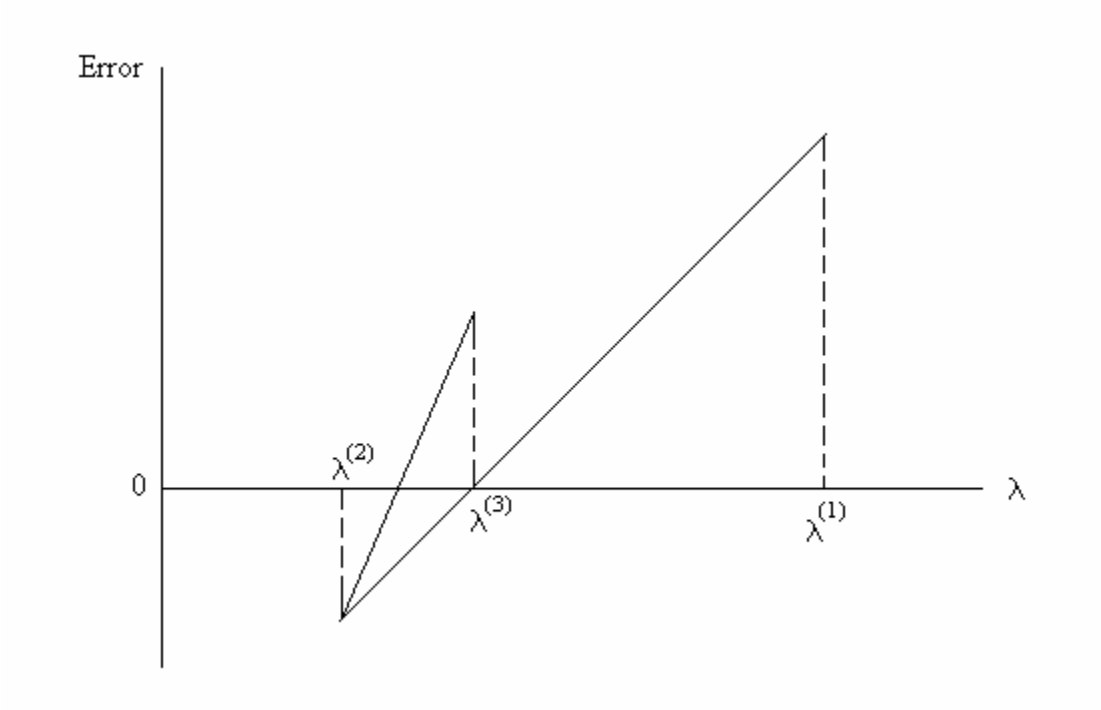
Considerando λ fijo se resuelven las ecuaciones anteriores y se obtienen valores para las potencias.

Se determina un *Error* mediante la diferencia entre la suma de las potencias obtenidas en la iteración y la demanda, si el *Error* es positivo, la siguiente iteración se realiza con un valor de λ menor al de la iteración anterior, si el *Error* es negativo, la iteración siguiente se realiza con un valor de λ mayor. Es decir:

- Si $Error^{(k)} = \sum_{i=1}^{N} P_i^{(k)} - P_{load} > 0 \quad \Rightarrow \quad \lambda^{(k+1)} < \lambda^{(k)}$

- Si $Error^{(k)} = \sum_{i=1}^{N} P_i^{(k)} - P_{load} < 0 \quad \Rightarrow \quad \lambda^{(k+1)} > \lambda^{(k)}$

Si entre dos iteraciones consecutivas el *Error* cambia de signo, el valor de λ para la iteración siguiente se obtiene mediante una interpolación en un gráfico del *Error* v/s λ, en la intersección de esta interpolación con el *Error* nulo.



El proceso iterativo se detiene cuando $\left|Error^{(k)}\right| < \xi$ para ξ suficientemente pequeño.

Método de Newton

El método de Newton corresponde a un método iterativo mediante el cual se obtiene la raíz de una función continua y diferenciable.

Este es uno de los métodos más eficientes para aproximar las soluciones de la ecuación $f(x) = 0$. El método de Newton empieza con una aproximación inicial x_0, la siguiente aproximación x_1 corresponde a la intersección con el eje x de la recta tangente a la gráfica de f en $(x_0, f(x_0))$. La aproximación x_2 corresponde a la intersección con el eje x de la tangente a la gráfica de f en el punto $(x_1, f(x_1))$, y así sucesivamente. Este proceso genera una sucesión $\{x_n\}$, definida por:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

La expresión anterior puede derivarse a partir de un desarrollo en serie de Taylor.

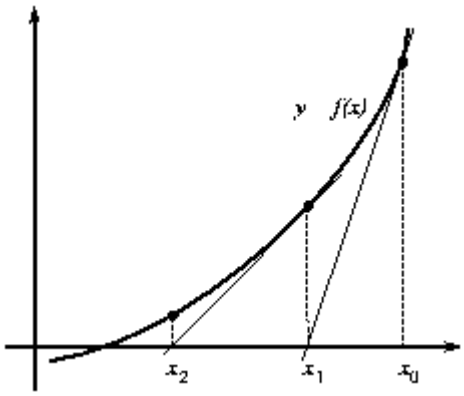
Efectivamente, sea r un cero de f y sea x una aproximación a r tal que $r = x + h$. Si f' existe y es continua, por el teorema de Taylor tenemos:

$$0 = f(r) = f(x+h) = f(x) + h \cdot f'(x) + O(h^2)$$

en donde $h = r - x$. Si x está próximo a r (es decir h es pequeña), es razonable ignorar el término $O(h^2)$:

$$f(x) + h \cdot f'(x) \quad \Rightarrow \quad h = -\frac{f(x)}{f'(x)}$$

El método de Newton tiene una interpretación geométrica sencilla, como se puede apreciar del análisis de la figura siguiente:



El método de Newton consiste en una linealización de la función, es decir, f se reemplaza por una recta tal que contiene al punto $(x_0, f(x_0))$ y cuya pendiente coincide con la derivada de la función en el punto, $f'(x_0)$. La nueva aproximación a la raíz, x_1, se obtiene de la intersección de la función linear con el eje X de ordenadas.

Como se trata de un método que busca un cero de la función, la convergencia se obtiene cuando se obtiene x_k tal que $f(x_k) \leq \xi$ para ξ suficientemente pequeño.

En el caso de despacho económico la función a la que se aplica el método corresponde al gradiente de la función lagrangeana.

$$f = \nabla L$$

F. ANEXO 6: RESULTADOS DEL PREDESPACHO DE UNIDADES DEL SING

El predespacho del SING mediante los algoritmos programados entregaron los siguientes resultados:

Algoritmo Branch and Bound

Generación Unidades en MW								
Unidad / Etapa	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CTM1 (Carbón)	154	154	129	126	116	104	124	131
CTM2 (Carbón)	163	163	163	163	163	163	163	163
CTTAR (Carbón)	140	100	100	100	100	100	100	100
NTO1 (Carbón)	127	127	127	127	127	127	127	127
NTO2 (Carbón)	131	131	131	131	131	131	131	131
TG11 + 0.5 TV10	300	300	300	300	300	300	300	300
TG2A + TG2B + TV2C (Diesel)	0	0	0	0	0	0	0	0
U10	0	0	0	0	0	0	0	0
U11	0	0	0	0	0	0	0	0
U12 (Carbón)	0	0	0	0	0	0	0	0
U13 (Carbón)	51	50	50	50	50	50	50	50
U14 (Carbón)	119	119	119	119	119	119	119	119
U15 (Carbón)	121	121	121	121	121	121	121	121
U16-TG + U16-TV	320	320	320	320	320	320	320	320
Total	**1626**	**1585**	**1560**	**1557**	**1547**	**1535**	**1555**	**1562**

Generación Unidades en MW								
Unidad / Etapa	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
CTM1 (Carbón)	103	114	135	128	110	90	90	101
CTM2 (Carbón)	163	163	163	163	163	152	157	163
CTTAR (Carbón)	100	100	100	100	100	100	100	100
NTO1 (Carbón)	127	127	127	127	127	127	127	127
NTO2 (Carbón)	131	131	131	131	131	131	131	131
TG11 + 0.5 TV10	300	300	300	300	300	300	300	300
TG2A + TG2B + TV2C (Diesel)	0	0	0	0	0	0	0	0
U10	0	0	0	0	0	0	0	0
U11	0	0	0	0	0	0	0	0
U12 (Carbón)	0	0	0	0	0	0	0	0
U13 (Carbón)	50	50	50	50	50	50	50	50
U14 (Carbón)	119	119	119	119	119	119	119	119
U15 (Carbón)	121	121	121	121	121	121	121	121
U16-TG + U16-TV	320	320	320	320	320	320	320	320
Total	**1534**	**1545**	**1566**	**1559**	**1541**	**1510**	**1515**	**1532**

Generación Unidades en MW								
Unidad / Etapa	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
CTM1 (Carbón)	140	138	154	154	154	154	154	154
CTM2 (Carbón)	163	163	163	163	163	163	163	163
CTTAR (Carbón)	100	100	133	135	139	140	140	126
NTO1 (Carbón)	127	127	127	127	127	127	127	127
NTO2 (Carbón)	131	131	131	131	131	131	131	131
TG11 + 0.5 TV10	300	300	300	300	300	300	300	300
TG2A + TG2B + TV2C (Diesel)	0	0	0	0	0	0	0	0
U10	0	0	0	0	0	0	0	0
U11	0	0	0	0	0	0	0	0
U12 (Carbón)	0	0	0	50	50	50	50	50
U13 (Carbón)	50	50	50	50	50	79	56	50
U14 (Carbón)	119	119	119	119	119	119	119	119
U15 (Carbón)	121	121	121	121	121	121	121	121
U16-TG + U16-TV	320	320	320	320	320	320	320	320
Total	**1571**	**1569**	**1618**	**1670**	**1674**	**1704**	**1681**	**1661**

Algoritmo de Relajación Lagrangeana

Generación Unidades en MW								
Unidad / Etapa	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CTM1 (Carbón)	145	104	90	90	90	90	90	90
CTM2 (Carbón)	163	163	152	149	139	127	147	154
CTTAR (Carbón)	100	100	100	100	100	100	100	100
NTO1 (Carbón)	127	127	127	127	127	127	127	127
NTO2 (Carbón)	131	131	131	131	131	131	131	131
TG11 + 0.5 TV10	300	300	300	300	300	300	300	300
TG2A + TG2B + TV2C (Diesel)	0	0	0	0	0	0	0	0
U10	0	0	0	0	0	0	0	0
U11	0	0	0	0	0	0	0	0
U12 (Carbón)	50	50	50	50	50	50	50	50
U13 (Carbón)	50	50	50	50	50	50	50	50
U14 (Carbón)	119	119	119	119	119	119	119	119
U15 (Carbón)	121	121	121	121	121	121	121	121
U16-TG + U16-TV	320	320	320	320	320	320	320	320
Total	**1626**	**1585**	**1560**	**1557**	**1547**	**1535**	**1555**	**1562**

Generación Unidades en MW								
Unidad / Etapa	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
CTM1 (Carbón)	90	90	90	90	90	90	90	90
CTM2 (Carbón)	126	137	158	151	133	102	107	124
CTTAR (Carbón)	100	100	100	100	100	100	100	100
NTO1 (Carbón)	127	127	127	127	127	127	127	127
NTO2 (Carbón)	131	131	131	131	131	131	131	131
TG11 + 0.5 TV10	300	300	300	300	300	300	300	300
TG2A + TG2B + TV2C (Diesel)	0	0	0	0	0	0	0	0
U10	0	0	0	0	0	0	0	0
U11	0	0	0	0	0	0	0	0
U12 (Carbón)	50	50	50	50	50	50	50	50
U13 (Carbón)	50	50	50	50	50	50	50	50
U14 (Carbón)	119	119	119	119	119	119	119	119
U15 (Carbón)	121	121	121	121	121	121	121	121
U16-TG + U16-TV	320	320	320	320	320	320	320	320
Total	**1534**	**1545**	**1566**	**1559**	**1541**	**1510**	**1515**	**1532**

Generación Unidades en MW								
Unidad / Etapa	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
CTM1 (Carbón)	90	90	138	154	154	154	154	154
CTM2 (Carbón)	163	161	163	163	163	163	163	163
CTTAR (Carbón)	100	100	100	136	139	140	140	126
NTO1 (Carbón)	127	127	127	127	127	127	127	127
NTO2 (Carbón)	131	131	131	131	131	131	131	131
TG11 + 0.5 TV10	300	300	300	300	300	300	300	300
TG2A + TG2B + TV2C (Diesel)	0	0	0	0	0	0	0	0
U10	0	0	0	0	0	0	0	0
U11	0	0	0	0	0	0	0	0
U12 (Carbón)	50	50	50	50	50	50	50	50
U13 (Carbón)	50	50	50	50	50	79	56	50
U14 (Carbón)	119	119	119	119	119	119	119	119
U15 (Carbón)	121	121	121	121	121	121	121	121
U16-TG + U16-TV	320	320	320	320	320	320	320	320
Total	**1571**	**1569**	**1619**	**1671**	**1674**	**1704**	**1681**	**1661**

G. ANEXO 7: UNIDADES GENERADORAS DEL SING

Unidades	Configuración	Potencia Bruta Máxima [MW]	Potencia Bruta Mínima [MW]
CC SALTA	TG11	208	115
	TG11 + 0.5 TV10	321.4	185
	TG11 + TG12	416	230
	TG11 + TG12 + TV10	642.8	365
	TG11, TG12+TV10	529.4	300
	TG12	208	115
	TG12 + 0.5 TV10	321.4	185
	TG12, TG11+TV10	529.4	300
CTTAR	CTTAR (Carbón)	158	100
TGTAR	TGTAR	23.75	8
CAVA	CAVA	2.602	0.5
CHAP	CHAP	10.2	1
GMAN	GMAN	16.8	2.1
MAAN	MAAN (Diesel)	11.872	5.936
GMAR	GMAR	8.4	2.1
M1AR	M1AR	2.997	0.999
M2AR	M2AR	2.924	1.462
MAIQ	MAIQ	5.936	5.936
MIIQ	MIIQ	2.924	1.462
MSIQ	MSIQ	6.2	6.2
SUIQ	SUIQ	4.2	1.4
TGIQ	TGIQ	23.75	10
CTM1	CTM1 (Carbón)	165.9	90
CTM2	CTM2 (Carbón)	175	90
CTM3	CTM3-TG	156.3	100
	CTM3-TG + CTM3-TV	250.75	160
CUMMINS	CUMMINS	0.722	0.722
DEUTZ	DEUTZ	1.959	0.653
MIMB	MIMB	28.64	2
TG1	TG1	24.698	10
TG2	TG2	24.931	10
TG3	TG3 (Diesel)	37.5	10
U10	U10	37.5	15
U11	U11	37.5	15
U12	U12 (Carbón)	85.3	50
U13	U13 (Carbón)	85.5	50
U14	U14 (Carbón)	128.3	75
U15	U15 (Carbón)	130.3	75
U16	U16-TG	280	75
	U16-TG + U16-TV	400	202

Unidades	Configuración	Potencia Bruta Máxima [MW]	Potencia Bruta Mínima [MW]
CC1	TG1A	126.74	95
	TG1A + 0.5 TV1C	194.39	155.1
	TG1A + TG1B	253.5	190
	TG1A + TG1B + TV1C	395.9	310
	TG1A, TG1B + 0.5 TV1C	321.13	250.1
	TG1B	126.74	95
	TG1B + 0.5 TV1C	194.39	155.1
	TG1B, TG1A + 0.5 TV1C	321.13	250.1
CC2	TG2A	123.71	95
	TG2A + 0.5 TV2C	189.09	155
	TG2A + TG2B	247.76	190
	TG2A + TG2B + TV2C	384.7	310
	TG2A, TG2B + TV2C	315.41	250
	TG2B	124.05	95
	TG2B + 0.5 TV2C	191.7	155
	TG2B, TG2A + TV2C	313.14	250
ZOFRI_1-6	ZOFRI_1-6	0.9	0.36
ZOFRI_2-5	ZOFRI_2-5	5.16	1.03
NTO1	NTO1 (Carbón)	136.3	65
NTO2	NTO2 (Carbón)	141.04	65