ISSN 0717-9103

META-RAPS: UN ENFOQUE DE SOLUCIÓN EFICAZ PARA PROBLEMAS COMBINATORIOS

META-RAPS: AN EFFECTIVE SOLUTION APPROACH FOR COMBINATORIAL PROBLEMS

REINALDO J. MORAGA [1]
GARY E. WHITEHOUSE [2]
GAIL W. DEPUY [3]

1 Universidad del Bío-Bío, Concepción - Chile
2 University of Central Florida – EE.UU.
3 University of Louisville – EE.UU

Resumen

Este artículo introduce una metaheurística denominada Meta-RaPS (Meta-heuristic for Randomized Priority Search) para problemas de optimización combinatoria. Meta-RaPS es un sistema de múltiples iteraciones que balancea el uso de heurísticas de construcción y mejoramiento de soluciones en cada iteración. Una de las características principales de Meta-RaPS es la dosificación del uso de aleatoriedad como mecanismo para mejorar heurísticas de construcción. En este artículo se presenta el enfoque y se entregan resultados de aplicaciones a cuatro problemas de optimización combinatoria. Este artículo es un extracto de la tesis doctoral titulada: *"Meta-RaPS: An Effective Solution Approach for Combinatorial Problems"* (Moraga, 2002). La tesis representa la culminación de una investigación desarrollada por las Universidades de Central Florida y Louisville en un esfuerzo por extender un enfoque heurístico clásico denominado COMSOAL a problemas combinatorios. (Nota: esta tesis doctoral es una de las tres tesis que recientemente obtuvieron el Premio *"2003 Pritsker Doctoral Dissertation Award"*, otorgado por el Institute of Industrial Engineering en la última Conferencia de Investigación en Ingeniería Industrial celebrada en Portland, Oregon, USA, Mayo 2003.)

Palabras Claves
Metaheurísticas, Optimización Combinatoria

1. Introducción

En el campo de la Investigación de Operaciones, numerosos problemas pueden ser estructurados de tal forma que optimicen una función objetivo que depende de algunas variables de decisión, las que a su vez están definidas sobre una región formada por un conjunto de restricciones. En términos matemáticos, un problema de optimización general puede ser formulado como:

Optimizar $f(x)$

$$\text{sujeto a } g_i(x)\begin{Bmatrix}\geq\\=\\\leq\end{Bmatrix}b_i; \quad i = 1,\text{K}\ ,m; \qquad (1)$$

$$x \geq 0$$

Donde $f(\cdot)$ es una función objetivo que debe ser optimizada sobre una región formada por la intersección de una familia de m restricciones $g_i(\cdot)$ y donde el vector x, compuesto por n variables de decisión, toma valores mayores o iguales a cero. El tipo de optimización puede ser de maximización o minimización. En el caso de maximización, el problema representado por (1) tiene solución cuando existe un vector x* para el cual $f(x^*)>f(x)$, donde x es cualquier otro vector dentro de la región definida por las restricciones $g_i(\cdot)$. A partir de la estructura básica representada arriba, una amplia gama de clasificaciones bastante conocidas en literatura pueden derivarse. Por ejemplo, asumiendo que las variables son continuas y las funciones f y g_j dependen linealmente del vector x, el problema se dice ser de optimización *lineal*, de lo contrario se conoce como de optimización *no-lineal*. En cambio, si las variables son de naturaleza discreta, el problema de encontrar soluciones óptimas es conocido como de optimización *combinatoria*. Además, cuando existe más de un objetivo compitiendo por recursos se tienen diversas funciones que deben ser optimizadas dando lugar a la optimización *multi-objetivo*. Por otro lado, un problema de optimización puede ser caracterizado por la certeza que se tenga de las variables. En el caso de que éstas sean estocásticas, una distribución de probabilidad se asocia a cada variable, en cuyo caso se habla de optimización *estocástica*.

Variadas son las aplicaciones de los problemas de optimización combinatoria en la industria. Algunos ejemplos incluyen: problemas de asignación de recursos, de balanceo de líneas, de ruteo y programación de vehículos, de planificación de instalaciones, de distribución de plantas, de secuenciación de trabajos y programación de máquinas, de planificación de la fuerza de trabajo, de distribución y logística, entre muchos otros. Yu (1998) edita un interesante libro con una colección de artículos reportando aplicaciones industriales de problemas de optimización combinatoria. Moraga & AlMazid (2000) discuten diversos casos de problemas combinatorios encontrados en la industria los cuales pueden ser modelados usando el problema del vendedor viajero (TSP). Algunos ejemplos serían la programación de trabajos en que los tiempos de preparación de máquinas dependen de la secuencia, la inserción de partes componentes en placas de circuitos integrados, secuencias de perforaciones para robots, reparto de cargas, máquinas de medición de coordenadas, rayos-x, diseño, etc.

La mayor parte de los problemas de optimización combinatoria pueden ser clasificados como NP-complete, una clase de problemas para los que no existe un algoritmo de tiempo polinomial que pueda resolverlos a optimalidad. Esto ha llevado a muchos investigadores a explorar diversos métodos para abordarlos. La mayoría de estos métodos puede ser ampliamente clasificados ya sea como algoritmos “exactos” o “heurísticas” (Aarts y Lenstra, 1993.) Algoritmos exactos son aquellos que producen una solución óptima y para ello emplean varias técnicas a objeto de reducir el espacio de búsqueda. Estos métodos exactos incluyen aquellos basados en técnicas tales como: ramificación y acotamiento, planos cortantes, y también programación lógica de restricciones. Estos algoritmos son razonablemente eficientes para problemas de tamaño modesto (Ignizio y Cavalier, 1994.) Aunque con ellos es posible en principio resolver problemas de cualquier tamaño, en la práctica no es así debido al gran número de soluciones posibles para cualquier problema de tamaño razonable.

Durante los años sesenta, los investigadores trataban de responder la siguiente pregunta: ¿Existe un algoritmo de optimización con tiempo de ejecución polinomial para un problema como el TSP? Hasta ahora, nadie ha podido encontrar una respuesta a esta pregunta. Sin embargo, Karp (1972) mostró que si la respuesta es “si” para el TSP, hay también un número de otros problemas difíciles para los cuales un algoritmo polinomial podría ser encontrado. Como ningún algoritmo ha sido aún encontrado para alguno de estos problemas, Reeves (1996) dice que esto sugiere categóricamente que la respuesta a la pregunta original es “no.” Por lo mismo, es que el área de optimización combinatoria resulta cada vez más atrayente para investigadores y académicos, ya que cualquier contribución en este ámbito tiene repercusiones directas en la industria.

Por la razón anterior es que la atención se ha centrado más y más en el uso de heurísticas. Reeves (1996) define el termino heurística de la siguiente forma: *“Una técnica heurística (o simplemente una heurística) es un método que busca buenas soluciones (i.e. soluciones cercanas al óptimo) a un costo computacional razonable sin poder garantizar optimalidad.”* A diferencia de un enfoque algorítmico, un método heurístico no tiene una base de matemática formal, es desarrollado más o menos por intuición, y no puede garantizar una solución óptima exacta (Ignizio & Cavalier, 1994.) En general, hay dos tipos de heurísticas fácilmente distinguibles para usar en problemas combinatorios, éstas son: de construcción y

de mejoramiento. Una heurística de construcción produce una solución factible como resultado final, en tanto que una heurística de mejoramiento comienza a partir de una solución factible y luego la mejora (Reinelt, 1994).

Las metaheurísticas (también llamadas heurísticas modernas) han aparecido durante las últimas dos décadas. El desarrollo de metaheurísticas, junto al avance de las tecnologías de información que proveen de computadores cada vez más rápidos, han permitido a investigadores y profesionales de la industria resolver problemas combinatorios complejos y de gran escala (Yu, 1998.) En general, estas meta-heurísticas toman inicialmente una solución factible, la cual es luego mejorada usando heurísticas de mejoramiento embebidas en una estructura más general, i.e., Simulated Annealing (SA), Algoritmos Genéticos (GA), Búsqueda Tabú (TS), y Redes Neuronales (NN). La característica común de estos enfoques es el uso de mecanismos para evadir óptimos locales.

Uno de estos mecanismos es el uso de aleatoriedad. Empíricamente, puede ser demostrado el hecho que dosificando cierto grado de aleatoriedad en las heurísticas al igual como combinándolas aleatoriamente permite obtener mejores soluciones para un problema de optimización combinatoria particular (Whitehouse & DePuy, 2001). Este artículo introduce un procedimiento general llamado *Meta-heuristic for Randomized Priority Search* (Meta-RaPS) que explota el hecho mencionado anteriormente. Meta-RaPS surge como consecuencia de una investigación cuyo fin era adaptar y extender las aplicaciones del enfoque COMSOAL (Computer Method of Sequencing Operations for Assembly Lines) a problemas combinatorios. Las secciones siguientes en este artículo presentan un extracto de la investigación aludida. Algunas de las exitosas aplicaciones de Meta-RaPS a un número de problemas de optimización combinatoria se encuentran en DePuy & Whitehouse (2000), DePuy & Whitehouse (2001), Moraga (2002), DePuy, Moraga, & Whitehouse (2003), DePuy, Moraga & Whitehouse (2003) y Moraga, DePuy, & Whitehouse (2003). Meta-RaPS es un enfoque prometedor que obtiene soluciones de muy buena calidad y su simplicidad lo hace ideal para profesionales de la industria. Una discusión más detallada y extendida sobre los diferentes tópicos que aquí se abordarán se encuentra en Moraga (2002).

2. Breve Revisión Sobre Meta-Heurísticas

Glover & Laguna (1997) definen el término "metaheurística" como *"una estrategía maestra que guía y modifica otras heurísticas para producir soluciones más allá de aquellas que son normalmente generadas en una solicitud por optimalidad local. Las heurísticas guiadas por tal meta-estrategia pueden ser procedimientos de alto nivel o nada más que una descripción de movidas disponibles para transformar una solución en otra, junto con reglas de evaluación asociadas"* (pág. 17). También ellos poponen un método de clasificación para metaheurísticas en términos de tres alternativas de diseño básicas: (1) el uso de memoria adaptativa, (2) el tipo de exploración de vecindario usado, y (3) el número de soluciones que se llevan de una solución a otra. Estas opciones son puestas en un esquema de la forma $\alpha|\beta|\gamma$, donde las elecciones para α son A (si la metaheurística tiene memoria adaptativa) y M (si el método es sin memoria). Las elecciones para β son N (para un método usando búsqueda de vecindario) y S (para un método usando muestreo aleatorio). Finalmente, γ puede ser 1 (si el método hace una movida en cada iteración) o P (para un enfoque basado en poblaciones de tamaño P). A continuación se presenta una revisión breve sobre las diversas metaheurísticas existentes.

2.1 Simulated Annealing

Las ideas básicas de Simulated Annealing (SA) están basadas en una investigación desarrollada por Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller (1953). Esencialmente, el algoritmo de Metropolis simula el enfriamiento y proceso de temple en metales. Kirkpatrick, Gelatt, & Vecchi (1983) y Cerny (1985) tomaron estas ideas y propusieron este enfoque para problemas de optimización. Desde entonces, matemáticos, estadísticos, físicos y científicos de la computación han estudiado extensivamente SA y la han aplicado a numerosas otras áreas. Algunos artículos que presentan excelentes revisiones así también como bibliografías descriptivas sobre el enfoque SA son Collins, Eglese, & Golden (1988), Tovey (1988), y Eglese (1990).

SA fue primero considerada por algunos autores como sólo una heurística para búsqueda local o de vecindario pero más sofisticada (Reeves, 1993; Eglese, 1990), en tanto que otros han convenido en clasificarla como metaheurística (Van Breedam, 2001; Glover & Laguna, 1997). De acuerdo al esquema de clasificación global que Glover y Laguna (1997) han propuesto, SA puede ser clasificada como ya sea una metaheurística M/S/1 o M/N/1. En otras palabras, SA es un enfoque sin memoria que podría emplear ya sea búsqueda de vecindario o muestreo aleatorio a medida que se mueve de una solución a otra.

Reinelt (1994) declara que SA puede entregar resultados de muy buena calidad, pero el tiempo de corridas puede ser largo a medida que la temperatura va lentamente disminuyendo y muchas repeticiones en cada paso de temperatura son necesarias. El mismo autor hace la observación de que la elección apropiada del esquema de enfriamiento es crítica, ya que es altamente dependiente del problema y sujeta a numerosos experimentos.

Numerosas aplicaciones de SA en el campo de la optimización combinatoria puede ser encontrado en la literatura, por ejemplo en conocidos problemas como: TSP, Problema de Ruteo de Vehículos (VRP), Problema de Asignación Cuadrática (QAP), Graph Coloring (GCP), y Problema de Programación de Máquinas (MSP).

2.2 Tabu Search

Tabu Search (TS) fue propuesta por Glover (1986). La filosofía detrás de TS es derivar y explotar una colección de principios para la solución inteligente de problemas. Un elemento subyacente a TS es el uso de memoria flexible, lo que involucra el proceso dual de crear y explotar estructuras para aprovechar la historia mientras se busca una solución (Glover & Laguna, 1997). Con lo anterior, a partir de una solución inicial, TS utiliza una estructura de vecindario local para derivar inteligentemente una solución cercana al óptimo.

Como meta-heurística, TS puede ser clasificada ya sea como A/N/1 o A/N/P (Glover & Laguna, 1997), lo cual significa que tiene memoria adaptativa usando búsqueda de vecindario y se mueve de una solución actual a la siguiente en cada iteración o después de un enfoque basado en poblaciones. En general, Reinelt (1994) declara que las dificultades básicas de esta metaheurística son el diseño de una lista tabú razonable, el manejo eficiente de la lista, y la selección de la movida más apropiada que no sea prohibida.

Tabu Search es la más reconocida entre las metaheurísticas y ha sido extensamente aplicada a numerosos problemas combinatorios tales como: VRP, TSP, QAP, y 0-1MKP. De acuerdo a Laporte, Gendreau, Potvin, & Semet (2000), TS ha sido la más exitosa meta-heurística, especialmente para resolver el problema VRP. En su libro, Glover y Laguna (1997) presentan una muy buena discusión sobre las aplicaciones de Tabu Search.

2.3 Algoritmos Genéticos

Los Algoritmos Genéticos (GA) fueron introducidos por Holland (1975) con la idea de imitar los procesos de selección y evolución natural. El enfoque de GA simula el proceso cuando las especies se adaptan a un ambiente complejo y cambiante a objeto de maximizar su probabilidad de sobrevivencia. El conocimiento que las especies adquieren se codifica en sus cromosomas y se transmite a la siguiente generación de especies, las cuales tendrán mayores probabilidades de sobrevivir. GA pueden ser clasificado como una metaheurística A/S/P (Glover & Laguna, 1997).

Schaffer & Eshelman (1996) reconocen que los GA no son la mejor alternativa en el caso de problemas combinatorios. Radcliffe & Surry (1995) dicen que esto se debe a que las representaciones que inducen a la obtención de buenos esquemas de cruzamiento son difíciles de encontrar. Este problema es maginificado más aún por el hecho de que todavía no existe una clara teoría de esquemas de cruzamiento (Radcliffe, 1990). La posibilidad de encontrar muy buenas soluciones cercanas al óptimo depende del tamaño del problema y calidad de la población inicial. Adicionalmente, Chatterjee, Carrera, & Lynch (1996) dicen que para ciertos problemas el cruzamiento puede conducir a la pérdida de información

por accidente. También, Reinelt (1994) dice que encontrar soluciones óptimas podría requerir considerable tiempo de CPU.

Se pueden encontrar variadas aplicaciones de GA a problemas combinatorios tales como: TSP, Secuenciamiento y programación, problema de la mochila, GCP, Empaquetado de Cajas (BPP), y VRP.

2.4 Redes Neuronales

Es bien sabido que las todas las funciones biológicas, incluyendo memoria, son almacenadas en neuronas y en las conexiones entre ellas. Además, el aprendizaje puede ser visto como el establecimiento de nuevas conexiones entre neuronas o modificaciones de conexiones existentes. Las neuronas biológicas han sido modeladas usando redes neuronales artificiales (RNA), lo cual ha llegado a ser una contribución muy original en el área de inteligencia artificial. Una RNA es básicamente un conjunto de capas conteniendo un número de nodos. Cada nodo de una capa particular recibe entradas desde nodos que pertenecen a capas precedentes y envía salidas a todos los nodos que pertenecen a las capas siguientes. Como resultado, se obtiene una red completamente interconectada. Usando el esquema de clasificación global que Glover y Laguna (1997) han propuesto, una RNA puede ser clasificada como una metaheurística A/N/1. En otras palabras, una RNA es un enfoque adaptativo que podría emplear una búsqueda de vecindario a medida que se mueve de una solución a otra. Aunque las aplicaciones de RNA han sido enfocadas en su mayor parte a diagnosis, predicción, clasificación, reconocimiento de patrones, y control; aplicaciones a problemas de optimización pueden también ser encontradas. Excelentes contribuciones son las de Gulati & Iyengar (1987), y Arizona, Yamamato, & Ohto (1992) quienes desarrollan una RNA para resolver el problema de programación de una máquina. En ambos casos son usadas algunas extensiones de la RNA de Hopfield, la cual es usada para abordar problemas de optimización (Hopfield & Tank, 1985). También Sabuncuoglu & Hommertzheim (1992) usan redes con propagación hacia atrás para resolver el problema de minimización de tardanzas. Bernard, De la Croix, & Le Texier (1988) propone mapas auto-organizados para resolver el problema TSP, su enfoque es basado en el trabajo de Kohonen (1984). Sin embargo, Reinelt (1994) argumenta que los resultados de RNA no son todavía convincentes ya que las rutas generadas son similares a las rutas de inserción más cercana con respecto a estructura y calidad.

Cada una de las metaheurísticas anteriormente presentadas no balancean las fases de construcción y mejoramiento. Es decir, la fase de construcción de soluciones factibles es mínima en comparación a una larga fase de mejoramiento a través de heurísticas de búsqueda local. Por lo general, estas metaheurísticas proponen generar aleatoriamente una solución factible en la fase de construcción. Aún GA que toma ventaja de un excelente paradigma aprovechando mecanismos de paralelismo es pobre en la fase de construcción. Algunos autores sugieren que es necesario en la generación de la primera solución incorporar mayor conocimiento del problema mediante el uso de una heurística de construcción. El balanceo de ambas fases de construcción y mejoramiento es una característica importante de Meta-RaPS.

3. Desde COMSOAL a Meta-RaPS

COMSOAL (Computer Method of Sequencing Operations for Assembly Lines) es una heurística originalmente reportada como enfoque de solución al problema de balanceo de líneas de ensamble (Arcus, 1966). Sin embargo, el concepto detrás de esta heurística puede ser aplicado a una variedad de problemas. Para un problema de optimización combinatoria en general, COMSOAL construye soluciones generando una lista de aquellas actividades que están disponibles para ser programadas. Nótese que las actividades deben ser factibles de programar para que ellas aparezcan en esta lista de disponibilidad (i.e. la actividad no está programada, las actividades predecesoras han sido terminadas, existen recursos disponibles, etc.) En su forma original, COMSOAL elige aleatoriamente la siguiente actividad a ser programada desde la lista de actividades disponibles. Este enfoque permite generar rápidamente soluciones de muy buena calidad.

Meta-RaPS significa *Meta-heuristic for Randomized Priority Search.* La filosofía detrás de de Meta-RaPS está basada sobre las ideas de Arcus previamente presentadas. Meta-RaPS integra reglas de prioridad (heurísticas), aleatoriedad y muestreo. En cada iteración, Meta-RaPS construye y mejora una solución factible mediante el uso de reglas de prioridad aleatorizadas. El procedimiento de Meta-RaPS modifica el enfoque de COMSOAL utilizando uno de dos criterios a medida que va construyendo una solución factible: selección aleatoria o selección basada en un esquema de prioridades. Una distribución definida por el usuario es empleada para determinar cómo será escogida la próxima actividad. Este proceso iterativo de seleccionar la próxima actividad a ser programada continua hasta que todas las actividades hayan sido programadas y por lo tanto una solución factible sea obtenida. Este hecho mejora la calidad de las soluciones factibles construidas. Adicionalmente, se incorpora una etapa de búsqueda local a la que una solución factible puede ser sometida. Esto se hace basándose en una distribución también definida por el usuario. Después de cierto número de iteraciones, Meta-RaPS conserva y reporta la mejor solución encontrada.

Empíricamente es posible demostrar las siguientes premisas que sustentan el procedimiento de Meta-RaPS:

- ❑ La incorporación de aleatoriedad en una heurística puede mejorarla dramáticamente.
- ❑ Las combinaciones aleatorias de heurísticas pueden conducir a resultados mejores que los que obtiene cada una de las heurísticas por separado.

En general, la aplicación de Meta-RaPS a cualquier problema de optimización combinatoria como metodología debe considerar los siguientes pasos:

1. Estudiar la estructura del problema.
2. Diseñar una heurística de construcción.
3. Modificar la heurística de construcción añadiendo aleatoriedad.
4. Diseñar fase de construcción combinando aletoriamente la heurística original y la modificada.
5. Diseñar fase de búsqueda local
6. Correr Meta-RaPS

Desde un punto de vista amplio, Meta-RAPS puede ser visto como una estrategia genérica para modificar heurísticas (o reglas de prioridad). Por esta razón, se clasifica en la categoría de metaheurísticas según la definición presentada arriba. Como en algunas otras metaheurísticas, el uso de aleatoriedad representa un mecanismo para evadir óptimos locales. Basados en la clasificación de Glover & Laguna, Meta-RAPS podría ser representada como una metaheurística M/S/1; i.e., sin memoria, usando muestreo aleatorio y se mueve de una solución a otra en cada iteración.

Meta-RaPS puede ser considerado una forma más general de COMSOAL como se ha mostrado anteriormente. Sin embargo, Meta-RaPS es también la forma general de otra metaheurística llamada GRASP (Greedy Randomized Adaptive Search Procedure; Feo & Resende, 1989). GRASP construye soluciones introduciendo aleatoriedad a una heurística de construcción avara mediante el uso de un parámetro de porcentaje de restricción en la misma forma como lo hace Meta-RaPS, Pero no considera la combinación aleatoria de reglas. GRASP también incluye mejoramiento mediante técnicas búsqueda local aplicada a todas las soluciones construidas. Meta-RaPS en cambio aplica técnicas de búsqueda local sólo aquellas soluciones que ofrecen mejor potencial de mejoramiento. De modo que Meta-RaPS puede imitar GRASP cuando no existe combinación de heurísticas y 100% de mejoramiento. Meta-RaPS ofrece mayor flexibilidad sobre COMSOAL y GRASP en que permite fijaciones de parámetros definidas por el usuario. Adicionalmente, GRASP usa mayormente heurísticas avaras mientras que Meta-RaPS has sido usado con heurísticas avaras y con otras más inteligentes (DePuy *et al.*, 2003, Moraga *et al.*, 2001). La calidad de soluciones que se obtiene muestra la flexibilidad extra abordada por la forma general de Meta-RaPS parece beneficiosa.

4. Meta-RaPS aplicado al Problema de la Mochila Multidimensional

Bastante atención ha sido dada al problema de la mochila multidimensional 0-1 (0-1MKP), el cual es bastante conocido. Esencialmente, este problema consiste de una mochila con *m* restricciones

limitadas por b_1, b_2, b_3,..., b_m, y n objetos, cada uno de ellos posee cierta utilidad c_i. El i^{th} objeto pesa aij cuando es considerado para una posible inclusión en la restricción jth de capacidad bj. En términos formales, el problema puede ser matemáticamente formulado como sigue:

Maximizar $$\sum_{i=1}^{n} c_i x_i \qquad (2)$$

Sujeto a $$\sum_{i=1}^{n} a_{ij}\, x_i \le b_j \qquad \forall j = 1,...,m \qquad (3)$$

$$x_i \in \{0,1\} \qquad \forall i = 1,...,n \qquad (4)$$

donde $$x_i = \begin{cases} 1 & si\ el\ objeto\ i\ es\ incluido\ en\ la\ mochila \\ 0 & de\ otro\ mod\,o \end{cases}$$

El problema 0-1MKP es una generalización del problema de la mochila 0-1 simple (para el cual m=1) y un caso especial de programación entera binaria. El caso unidimensional referido ha sido extensamente estudiado por Martello & Toth (1990), e incluso han sido propuestos algunos algoritmos de aproximación de tiempo polinomial por Ibarra & Kim (1975) y Sahni (1975). La importancia del problema 0-1MKP emerge tanto desde un punto de vista práctico como teórico. El problema 0-1MKP abarca muchas aplicaciones practicas, algunas de ellas incluyen problema de patrones de cortes (Gilmore & Gomory, 1966), problemas de carguío (Bellman, 1957; Shih, 1979), reparto de abarrotes en vehículos con múltiples compartimentos (Chajakis & Guignard, 1992), selección de proyectos (Petersen, 1967; Kyparisis, Gupta & Ip, 1996), presupuesto de capital (Weingartner, 1967), y asignación de procesadores y datos en sistemas computacionales distribuidos (Gavish & Pirkul, 1982.) Adicionalmente, problemas tales como cobertura de conjuntos pueden ser reformulados como un problema 0-1MKP equivalente mediante complementación de variables (Hochbaum, 1996; Bertsimas & Demir, 2002); en tanto que el conocido problema de ruteo de vehículos puede ser reducido al problema de la mochila 0-1 para algunas condiciones especiales (Eilon, Watson-Gabdy, & Christofides, 1971).

4.1 Heurística de construcción para el problema 0-1MKP

Muchas de las reglas de prioridad para el problema de la mochila 0-1 con restricciones múltiples (0-1MKP) son de tipo avaras basadas en el cálculo de una razón de pseudo-utilidad para cada objeto, i.e., $\alpha_i = c_i / w_i$, donde w_i es el factor de penalización (Loulou & Michaelides, 1979). Luego una lista de objetos es construida de acuerdo a la magnitud no creciente de sus razones de pseudo-utilidad. Mientras la lista no haya sido exhaustivamente revisada, el i^{th} objeto es asignado a la solución en construcción sólo si ninguna restricción es violada, de otro modo el objeto es descartado. Existen variadas formas de cómo calcular el factor de penalización. La más simples de todas, denominada SGR (simplest greedy rule), es la propuesta por Eilon, Watson-Gabdy, & Christofides (1971), la cual consiste en dividir todas las restricciones por sus correspondientes valores del lado derecho b_j's y luego sumarlas. De esta forma, la familia completa de restricciones se transforma en sólo una restricción equivalente donde el factor de penalización para el objeto i^{th} esta dado por la siguiente expresión:

$$w_i = \sum_{j=1}^{m} \left(\frac{a_{ij}}{b_j} \right)$$

4.2 Heurística de mejoramiento para el problema 0-1MKP

La etapa de mejoramiento implementada en Meta-RaPS básicamente consiste en dos técnicas de búsqueda de vecindario: inserción e intercambio. Primero, uno de los objetos no asignados a la solución factible es aleatoriamente seleccionado e insertado en la solución actual sin violar la factibilidad. Si esta movida es posible, la solución mejorará y el procedimiento va al siguiente objeto no asignado. Segundo, si la inserción no es posible, entonces el objeto no asignado ya elegido comienza a ser intercambiado por un objeto asignado que es aleatoriamente seleccionado. Si la factibilidad de la solución no es violada y el valor de la función objetivo mejora, la movida se mantiene. De otro modo, la movida es descartada y un nuevo objeto asignado es seleccionado aleatoriamente.

4.3 Funcionamiento

En Meta-RaPS, el parámetro %p define el porcentaje de veces que el siguiente objeto será elegido usando la regla SGR. Las veces restantes (i.e. 100-%p) el siguiente objeto es aleatoriamente elegido de entre aquellos objetos cuyas razones de pseudo-utilidad están dentro de un rango %r del objeto elegido por la regla SGR. Específicamente, si los parámetros %p=20 y %r=40 son considerados, entonces el 20% de las veces el objeto con la mejor razón de pseudo-utilidad es añadido a la mochila si la factibilidad no es violada. El restante 80% de las veces, el siguiente objeto es aleatoriamente escogido desde una lista con aquellos objetos cuyas razones de pseudo-utilidad son superiores al valor (1-0.40)*(mejor razón de pseudo-utilidad). Este método permite disminuir la posibilidad de quedar atrapados en algún óptimo local, y ayuda al usuario a encontrar una respuesta más cercana al óptimo global. Un parámetro adicional, porcentaje de mejoramiento %i, es usado para decidir qué soluciones construidas pasarán por la etapa de mejoramiento. Todas aquellas soluciones construidas cuyos valores de la función objetivo son superiores a (100-%i)*(mejor valor de solución no mejorada) pasarán por mejoramiento. A medida que corren las iteraciones, el mejor valor de solución no mejorada va siendo mantenido. Después de un número de iteraciones, la mejor solución sobre todas las iteraciones es reportada.

4.4 Resultados

La aplicación tal como ha sido descrita fue codificada en C++ y las corridas fueron hechas en un PC Pentium IV 1.6 GHz. Una biblioteca estándar de 56 problemas de prueba publicados en la literatura y disponibles en las páginas de OR-Library (Beasley, 2003) y MP-TESTDATA (Skorobohatyj, 1999) es usado para evaluar el enfoque Meta-RaPS. En los 56 problemas usados en este experimento, el número de restricciones m varía de 2 a 30 y el número de variables de 6 a 105 con valores óptimos conocidos. Los parámetros fueron fijados experimentalmente en %p=30, %r=50, %i=15, y el número de iteraciones I=10,000. Los resultados son mostrados en la Tabla 1 en terminos del número óptimo de soluciones encontradas, porcentaje de desviación desde la solución óptima, y los tiempos de corridas para Meta-RaPS SGR (Meta-RaPS basado en SGR) y SGR.

Tabla 1: Resultados de Meta-RaPS para 56 problemas 0-1 MKP

Técnica usada	# Soluciones óptimas	% desviación promedio desde el óptimo	Tiempo de corrida (seg/prob)
Meta-RaPS SGR	38/56	0.105%	12
SGR	4/56	3.711%	<0.1

Los resultados entregados por Meta-RaPS SGR son muy superiores a los obtenidos por la heurística SGR. Adicionalmente, los resultados de Meta-RaPS SGR fueron contrastados a otras conocidas técnicas usadas para resolver los mismos 56 problemas de prueba (algunos autores reportan el uso de 57 problemas), como se muestra en Tabla 2. La calidad de soluciones de la implementación Meta-RaPS SGR compara favorablemente a las implementaciones de Simulated Annealing y a algunas implementaciones de Tabu Search. Nótese que en la Tabla 3 además se reportan resultados de una

implementación reciente de Meta-RaPS basada en una heurística dinámica (Meta-RaPS DGR), la que permite mejorar la calidad de soluciones entregadas por Meta-RaPS, en este caso los resultados son bastante prometedores comparados a las otras técnicas. Mayores antecedentes obre esta nueva implementación son encontrados en Moraga, DePuy & Whitehouse (2003) .

Tabla 2. Comparación de Meta-RaPS a otras técnicas de solución

Técnicas de Solución	Soluciones óptimas	% desviación promedio desde el óptimo
Meta-RaPS SGR	38/56	0.105%
Meta- RaPS DGR (Moraga, DePuy & Whitehouse, 2003)	55/56	0.003%
Simulated annealing DETEXC (Drexel, 1988)	7/57	1.739%
Simulated annealing PROEXC (Drexel, 1988)	23/57	0.239%
Simulated annealing (Drexel, 1988 as implemented by Dammeyer and Voss, 1993)	31/57	0.328%
Tabu Search REM (Dammeyer & Voss, 1993)	40/57	0.126%
Tabu Search STM (Dammeyer & Voss, 1993)	39/57	0.130%
Tabu Search L+STM (Dammeyer & Voss, 1993)	44/57	0.101%
Tabu Search (Glover & Kochenberger, 1996)	57/57	0.000%
Tabu Search (Lokketangen & Glover, 1998)	37/54	0.003%
Genetic Algorithm (Chu & Beasley, 1998)	55/55	0.000%
Fix+Cut based method (Osorio *et al.*, 2003)	55/55	0.000%

5. Resultados de Meta-RaPS en Varios Problemas Combinatorios

En esta sección, se presenta un resumen de los resultados de Meta-RaPS obtenidos para cuatro problemas combinatorios, estos son: TSP, VRP, 0-1MKP, y RCPSP (problema de programación de proyectos con recursos restringidos.) La Tabla 3 muestra el porcentaje de desviación promedio desde el óptimo obtenido por Meta-RaPS en cada tipo de problema de optimización combinatoria considerado y el número de problemas de prueba usados para la experimentación. La primera fila muestra los resultados de Meta-RaPS, en tanto que la segunda fila presenta los resultados promedios al usar la heurística de prioridad por sí sola. Adicionalmente, la Tabla 3 compara la desviación promedio de Meta-RaPS desde el óptimo con respecto al rango del porcentaje de desviaciones promedios usando enfoques de solución de otros investigadores.

Tabla 3: Meta-RaPS versus otras técnicas en cuatro problemas de optimización combinatoria (% desviación promedio desde el óptimo)

	RCPSP	TSP	0-1MKP	VRP
Método de Solución	(110 problemas)	(5 problemas)	(56 problemas)	(3 problemas)
Meta-RaPS	0.35%	0.08%	0.11% (*)	1.38%
Regla de prioridad	5.00%	1.26%	3.71%	3.02%
Rango de enfoques de solución usados por otros investigadores	0.43%-2.95%	0.00%-8.26%	0.00%-3.05%	0.00%-5.11%

(*) corresponde a la implementación Meta-RaPS SGR

Los resultados de la Tabla 3 confirman las premisas declaradas inicialmente que sustentan el diseño de Meta-RaPS.

6. El Problema de Calibración de Parámetros

En general, uno de los problemas discutidos en la literatura involucra la necesidad de calibrar los parámetros de una metaheurística cuando se desempeña frente a un problema de optimización combinatoria (Van Breedman, 1995; Gendreau *et al.*, 1994). Por ejemplo, muchas de las metaheurísticas usadas para resolver el problema VRP contienen un gran número de parámetros (desde 5 a más de 25 parámetros en algunos casos) cuyos valores necesitan ser calibrados antes de correr la metaheurística. En el caso de Meta-RaPS existen sólo cuatro parámetros que calibrar: el porcentaje de restricción (%r), el porcentaje de prioridad (%p), el porcentaje de mejoramiento (%I), y el número de iteraciones (I). Sin embargo, a medida que %ie Iincrementan la solución tenderá siempre a ser mejor. Aún más, buenas solución construidas proveen buenas soluciones mejoradas, lo cual significa que el problema de calibración de parámetros en Meta-RaPS está asociado a los parámetros %py %$r$que controlan la calidad de las soluciones construidas. Por lo tanto, los experimentos realizados buscarán calibrar estos parámetros.

Los parámetros en Meta-RaPS pueden ser arbitrariamente fijados en base a algún grado de conocimiento del problema o de acuerdo a algún procedimiento más sistemático. En la metodología presentada a continuación se usan algunas ideas planteadas originalmente por Coy *et al.* (2001):

Paso 1. Seleccionar una muestra de problemas representativo de la colección de problemas.

Paso 2. Seleccionar el dominio de parámetros sobre el cuál éstos serán variados.

Paso 3. Seleccionar valores adecuados de los parámetros para cada problema en la muestra mediante el uso de una técnica para tal fin.

Paso 4. Combinar las calibraciones obtenidas en Paso 3 y correr la colección entera de problemas con la calibración obtenida.

El corazón de esta metodología se encuentra en el Paso 3 que consiste en encontrar la mejor calibración de parámetros para cada problema de la muestra escogida usando alguna técnica. Luego, usando una combinación de estos valores encontrados se corre toda la colección de problemas bajo estudio (Paso 4.) Dada la naturaleza estocástica de Meta-RaPS, el problema de calibrar sus parámetros puede ser abordado usando cualquiera de las técnicas ampliamente empleadas en el problema de optimización en simulación. Sin embargo, en este artículo se recomendará una de estas técnicas luego de comparar cuatro potenciales técnicas que pueden ser usadas para la fijación de parámetros. Las técnicas comparadas son: ordenamiento y selección, comparaciones múltiples, búsqueda estocástica mediante GA, y superficie de respuesta. El problema usado para mostrar el uso de cada una de estas técnicas y luego compararlas es el problema TSP, se considera una colección de cinco problemas KRO de 100 ciudades encontrados en TSPLIB (Reinelt and Bixby, 1995). El problema KROE se toma como la muestra representativa con la cual se aplicarán las cuatro técnicas.

La Tabla 4 muestra los resultados obtenidos después de utilizar las cuatro técnicas indicadas previamente para calibrar los parámetros de Meta-RaPS usando el problema TSP KROE de 100 ciudades. Con los resultados obtenidos con cualquiera de las técnicas anteriores se debería correr la colección completa de problemas TSP. Por ejemplo, usando la técnica de ordenamiento y selección, los valores de parámetros obtenidos para %p y %r serían 50 y 10 respectivamente que corresponde al Paso 3. Correspondería ahora, Paso 4, correr la colección de cinco problemas TSP usando los valores previamente obtenidos. Sin embargo, al comparar cuatro posibles técnicas que pueden ser usadas en Paso 3, se observa que la técnica de búsqueda estocástica con GA es la más prometedora de todas.

Table 4: Resultados de cuatro técnicas de calibración de parámetros en Meta-RaPS (Problema TSP KROE 100 ciudades)

	Técnicas de Calibración de Parámetros			
	Ordenamiento y Selección	Comparaciones múltiples	Búsqueda Estocástica GA	Superficie de Respuesta
Mejor calibración obtenida	*p*=50 *r*=10	*p*=99 *r*=22	*p*=97 *R*=39	*p*=92 *r*=47
Solución para *I*=1000	22349.90	22232.29	22167.23	22299.51
Desviación	1.28%	0.74%	0.45%	1.05%
Solución para *I*=5000	22120.14	22181.25	22110.82	22176.55
Deviación	0.24%	0.51%	0.19%	0.49%
Solución para *I*=10000	22106.33	22162.74	22106.33	22241.33
Deviación	0.17%	0.43%	0.17%	0.79%

Una comparación mayor de las cuatro técnicas es realizada corriendo Meta-RaPS en cada uno de los cinco problemas de la colección usando cada una de las cuatro calibraciones obtenidas previamente. La desviación promedio desde el óptimo obtenida para tres niveles de iteraciones, *I*, 1000, 5000, y 10000 es mostrada en la Figura 1. Se observa que la utilización de búsqueda estocástica con GA domina todas las otras técnicas en la colección de problemas TSP para cada uno de los niveles de iteraciones usados. Una discusión más detallada sobre este tópico en Meta-RaPS se encuentra en Moraga (2002).

7. Conclusiones y Recomendaciones

Este artículo ha presentado un extracto de los resultados de una investigación efectuada para modificar una heurística clásica denominada COMSOAL para resolver problemas combinatorios. El resultado ha sido el desarrollo de Meta-RaPS. Un número de heurísticas clásicas usadas para resolver conocidos problemas combinatorios, tales como: RCPSP, TSP, 0-1MKP, y VRP, han sido revisadas y aleatorizadas bajo el esquema de Meta-RaPS. La demostración del hecho que añadir cierto grado de aleatoriedad a una heurística (o regla de prioridad) mejora los resultados de la heurística original constituye un aporte importante de esta investigación. La aleatoriedad es también un mecanismo clave en otras metaheurísticas para eliminar la posibilidad de quedar atrapado en un óptimo local. El uso inteligente y dosificado de aleatoriedad como es el caso de Meta-RaPS permite obtener resultados competitivos en calidad y eficiencia frente a metaheurísticas como: GA, SA, TS, y NN.

Un aspecto importante a resolver cuando se usa cualquier metaheurística es el tema de calibración de parámetros, lo cual es indispensable para asegurar la calidad de soluciones. Laporte *et al.* (1999) declaran *"Por el lado algorítmico, ha llegado el momento de concentrarse en el desarrollo de algoritmos más robustos, más simples, y más rápidos, incluso si esto causa una pequeña pérdida en la calidad de las soluciones. Estos atributos son esenciales si queremos ver más de nuestros algoritmos implementados en paquetes comerciales"*. La explotación de estos atributos constituye en esencia una clara ventaja de Meta-RaPS sobre otros enfoques.

Meta-RaPS sin embargo es dependiente de la heurística embebida. Esto podría presentar un obstáculo para que Meta-RaPS funcione efectivamente en cualquier problema combinatorio, pero no es una desventaja tan clara en comparación a otras metaheurísticas. Entre mejor es la heurística de construcción para un problema particular, mejor será la calidad de solución utilizando Meta-RaPS. Sin embargo, la ausencia de heurísticas establecidas no impide la aplicación de Meta-RaPS, ya que la generación de soluciones bajo un esquema aleatorio es siempre posible.

El problema de calibrar parámetros en Meta-RaPS es una tarea de difícil dada la naturaleza estocástica del algoritmo. Cualquiera de las cuatro técnicas constrastadas permite manejar este problema, sin embargo la búsqueda estocástica con GA resulta prometedora con respecto a las otras técnicas. El uso de GA exibe la versatilidad de Meta-RaPS para la hiridización con otros enfoques interesantes.

Dos direcciones emergen a partir de este estudio para investigación futura en Meta-RaPS. Primero, es necesario continuar extendiendo estos resultados especialmente en aplicaciones más complejas y, segundo, buscar nuevos mecanismo que permitan mejorar aún más el enfoque Meta-RaPS. Una discusión más detallada sobre conclusiones y líneas futuras de investigación en Meta-RaPS se encuentra en Moraga (2002).

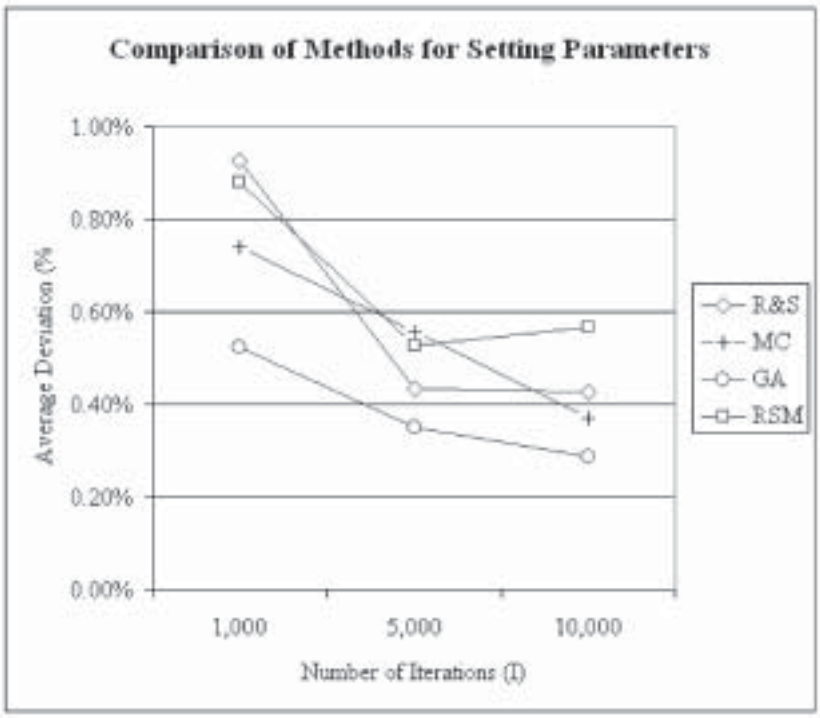

Figura 1: Comparación de Métodos para Calibrar Parámetros

8. Bibliografía

1) Aarts, E., & Lenstra, J. (1993). *Local Search in Combinatorial Optimization*. John Wiley & Sons.
2) Arcus, A.. (1966). COMSOAL: A Computer Method of Sequencing Operations for Assembly Lines, I The Problem in Simple Form. In Buffa E., *Readings in Production and Operations Management,* John Wiley & Sons.
3) Arizona, I., Yamamato, A., & Ohto, H., (1992), Scheduling for minimizing total actual flow time by neural networks, International Journal of Production Research 30, (3).
4) Beasley, J.E. (2003). OR-Library: Distributing Test Problems by Electronic Mail. *Journal of the Operational Journal Society*, 41, 170-181. <http://www.ms.ic.ac.uk/info.html>
5) Bellman R., (1957). *Dynamic Programming*. Princeton University Press, Princeton, NJ.
6) Bernard, A., De La Croix, G., & Le Texier, J. (1988). Self-organizing feature maps and the travelling salesman problem, *Neural Networks* v 1 n 4 1988 p 289-293.
7) Bertsimas, D., & Demir, R. (2002). An approximate dynamic programming approach to multidimensional knapsack problem. *Management Science*, 48 (4), 550-565.
8) Cerny, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient

simulation algorithm. *Journal of Optimization Theory and Applications* 45, 41-51.
9) Chajakis, E., & Guignard, M. (1992). A model for delivery of groceries in vehicle with multiple compartments and Lagrangean approximation schemes. In: Proceedings of Congreso Latino Ibero-Americano de Investigación de Operaciones e Ingeniería de Sistemas 1992, Mexico City.
10) Chatterjee, S., Carrera, C., & Lynch, L. (1996). Genetic algorithms and traveling salesman problems. *European Journal of Operational Research* 93, 490-510.
11) Chu, P.C., & Beasley, J.E. (1998). A genetic algorithm for the Multidimensional Knapsack Problem. *Journal of Heuristics*, 4, 63-86.
12) Collins, N., Eglese, R., & Golden B., (1988), Simulated annealing-An annotated bibliography, *American Journal of Mathematical and management Sciences* 8, 209-307
13) Coy, S., Golden, B., Runger, G., & Wasil E. (2001). Using Experimental Design to Find Effective Parameter Setting for Heuristics. *Journal of Heuristics*, 7:77-97.
14) Dammeyer, F., & Voss, S. (1993). Dynamic tabu list management using the reverse elimination method. *Annals of Operations Research*, 41, 31-46.
15) DePuy, G., & Whitehouse, G. (2000). Applying the COMSOAL Computer Heuristic to the Constrained Resource Allocation Problem. *Computers and Industrial Engineering* 38, 413-422.
16) DePuy, G., & Whitehouse, G. (2001). A Simple and Effective Heuristic for the Multiple Resource Allocation Problem. *International Journal of Production Research*, 32(4), 24-31.
17) DePuy, G., Whitehouse, G., & Moraga, R. (2003). Using The Meta-RaPS Approach To Solve Combinatorial Problems. Under review by *Computers and Industrial Engineering*.
18) DePuy, G.W., Moraga R.J., & Whitehouse, G.E., (2003). Meta-RaPS: A Simple and Effective Approach for Solving the Traveling Salesman Problem. In review by *Transportation Research Part B: Methodological.*
19) Drexel, A., (1988). A simulated annealing approach to the multiconstraint zero-one knapsack problem, *Computing*, 40, 1-8.
20) Eglese, R. (1990). Simulated Annealing: A tool for Operational Research. *European Journal of Operational Research* 46, 271-281.
21) Eilon, S., Watson-Gabdy, C., & Christofides, N. (1971). *Distribution Management: Mathematical modeling and practical analysis*. Hafner Publishing Company.
22) Feo, T., & Resende, M. (1995). Greedy randomized adaptive search procedures, *Journal of Global Optimization*, 6,109-133.
23) Gavish B., & Pirkul, H. (1982). Allocation of databases and processors in a distributed data processing. In: J. Akola, *Management of Distributed Data Processing* (pp. 215-231), North-Holland, Amsterdam.
24) Gendreau, M., Hertz, A., & Laporte, G. (1994). A Tabu Search Heuristic for the Vehicle Routing Problem. *Management Science*, 40, 1276-1290.
25) Gilmore, P.C., & Gomory, R.E. (1966). The theory and computation of knapsack functions, *Operations Research* 14, 1045-1074.
26) Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operation Research* 5, 533-549.
27) Glover, F., & Laguna, M. (1997). *Tabu Search*. Kluwer Academic Publishers.
28) Glover, F., & Kochenberger, G. (1996). Critical event tabu search for multidimentional knapsack problems, In: I.H. Osman & J.P. Kelly, *Meta-heuristics: Theory & Applications* (pp.407-427). Kluwer Academics Publishers.
29) Gulati, D., & Iyengar, S. (1987). Nonlinear networks for deterministic scheduling, Proceedings of the ICNN, 4.
30) Hochbaum, D.S. (1996). *Approximation Algorithms for NP-hard Problems.* New York: PWS Publishing Company.
31) Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.
32) Hopfield, J., & Tank, D. (1985). Neural computation of decisions in optimization problems. *Biological Cybernetics* n52, 141-152.
33) Ibarra, O.H, & Kim, C.E. (1975). Fast Approximate Algorithms for the 0/1 Knapsack Problem and Sum of Subset Problems. *Journal of the Association for Computing Machinery*, 22(4), 463-468.
34) Ingnizio, J., & Cavalier, T. (1994), *Linear Programming*. Prentice Hall.
35) Karp, R. (1972). Reducibility among combinatorial problems. In Miller R., and Thatcher J. (Eds), *Complexity of Computer Communications*. Plenum Press, New York.
36) Kirkpatrick, S., Gelatt Jr., C., & Vecchi, M. (1983). Optimization by Simulated Annealing, *Science* 222, 671-680.
37) Kohonen, T. (1984). Self-Organized Formation of Feature Maps, *Systems Science and Cybernetics*, 3-12.
38) Kyparisis, G., Gupta, S., & Ip, C. (1996). Project selection with discount returns and multiple

constraints. *European Journal of Operational Research*, 94(1), 87-96.
39) Laporte, G., Gendreau, M., Potvin, J., & Semet F. (2000). Classical and modern heuristics for the vehicle routing problem. *Int. Trans. in Op. Res*. 7, 285-300.
40) Laporte, G., Mercure, H., & Nobert, Y. (1992). A branch-and-bound algorithm for a class of asymmetrical vehicle routing problems. *Journal of Operational Research Society*, 43.
41) Lokketangen, A., & Glover, F. (1998). Solving zero-one mixed integer programming problems using tabu search, *European Journal of Operations Research*, 106, 624-658.
42) Loulou, R., & Michaelides, E. (1978). New Greedy-like Heuristics for the Multidimensional 0-1 Knapsack Problem, *Operations Research*, 27(6), 1101-1114.
43) Martello, S., & Toth, P. (1990). *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons.
44) Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., & Teller, E. (1953). Equation of State Calculation by Fast Computing Machines. *Journal Chem. Physics* 21, 1087-1092.
45) Moraga, R., & AlMazid, M. (2000). The traveling salesman problem (TSP) and its industrial applications. Proceeding of the 4th International Conference on Engineering Design and Automation, Orlando, FL.
46) Moraga, R., Whitehouse, G., DePuy, G., Neyveli, B., & Kuttuva, S. (2001). Solving the Vehicle Routing Problem Using the Meta-RaPS Approach, 29th International Conference on Computers and Industrial Engineering, November 1-3, Montreal, Canada.
47) Moraga, R.J, DePuy, G., & Whitehouse, G. (2003). Meta-RaPS Approach for Solving the 0-1 Multidimensional Knapsack Problem. Under review by Computers and Industrial Engineering.
48) Moraga, R.J., (2002). *Meta-RaPS: An Effective Solution Approach for Combinatorial Problems*. Ph.D. thesis. Orlando, FL: University of Central Florida.
49) Osorio, M.A., Glover, F., & Hammer, P. (2003). Cutting and Surrogate Constraint Analysis for Improved Multidimensional Knapsack Solutions. *Annals of Operations Research*, 117, 71-93.
50) Petersen, C.C. (1967). Computational experience with variants of the Balas algorithm applied to the selection of R&D projects. *Management Science*, 13(9), 736-750.
51) Radcliffe, N. (1990). *Genetic Neural Networks on MIMD Computers*. Ph.D. thesis. Edinburg, Scotland: Dept. of Theoretical Physics, University of Edinburg.
52) Radcliffe, N., & Surry, P. (1995). Fitness variance of formae and performance prediction. In Whitley D. and Vose M. (Eds), *Foundations of Genetic Algorithms* 3, pages 51-72. San Mateo, CA.
53) Reeves, C. (1993). *Modern Heuristic Techniques for Combinatorial Problems*, John Wiley & Sons.
54) Reeves, C. (1996). Modern Heuristic Techniques. In Rayward-Smith, I., *et al*. (Eds), *Modern Search Methods*, John Wiley & Sons.
55) Reinelt, G. (1994). The Traveling Salesman: Computational Solutions for TSP Applications. *Lectures Notes in Computer Science*, Springer-Verlag
56) Reinelt, G., & Bixby, W., (1995), TSPLIB--a library of travelling salesman and related problem instances. http://www.crpc.rice.edu/softlib/catalog/tsplib.html
57) Sabuncuoglu, I., & Hommertzheim, D. (1992). Artificial neural networks: Investigations and developments of neural network for scheduling problems. Presented at 1992 TIMS/ORSA Joint National Meeting, Orlando, 1992.
58) Schaffer, J., & Eshelman, L. (1996). Combinatorial Optimazation by Genetic Algorithms: The Value of the Genotype/Phenotype Distinction, In Rayward-Smith, I., *et al*. (Eds), *Modern Search Methods*, John Wiley & Sons.
59) Shih, W. (1979). A branch and bound method for the multiconstraint zero-one knapsack problem. *Journal of Operation Research Society*, 30, 369-378.
60) Skorobohatyj, G., (1999). MP-TESTDATA: Integer/Mixed-Integer Programming Problems. <http://elib.zib.de/pub/Packages/mp-testdata/ip/index.html>
61) Tovey, C. (1988). Simulated Annealing, *American Journal of Mathematical and Management Sciences*, 8, 389-407.
62) Van Breedam, A. (1995). Improvement Heuristics for the Vehicle Routing Problem Based on Simulated Annealing. *European Journal of Operation Research* 86, 480-490.
63) Van Breedam, A. (2001). Comparing descent heuristics and metaheuristics for the vehicle routing problem. *Computer and Operation Research* 28, 289-315.
64) Weingartner, H.M. (1967). *Mathematical Programming and the Analysis of Capital Budgeting Problems*. Chicago: Markham Publishing.
65) Whitehouse, G., & DePuy, G, (2001), Solving Constrained Multiple Resource Networks Both Forward and Backward Using Brooks Algorithm, *Project Management Journal*, Vol. 32, No. 4, pp. 24-31.
66) Yu, G. (1998). Industrial Applications of Combinatorial Optimization. Kluwer Academic Publishers.