Equipo asíncrono de agentes basados en recocido simulado aplicado al problema del agente viajero simétrico.

Asynchronous team of agents based on simulated annealing applied to solve the travel salesman problem

Luís Miguel Escobar[1], David Álvarez Martínez[2], Rubén Augusto Romero[3]
Universidad Tecnológica de Pereira, Pereira, Colombia
luismescobarf@gmail.com
akavallo@gmail.com
ruben@dee.feis.unesp.br

***Resumen*— En este artículo se presenta una metodología basada en A-Teams para resolver el problema del agente viajero. Para implementar el método de solución se usan como agentes diferentes versiones del Recocido Simulado. En el planteamiento del problema la función objetivo considera el costo del ciclo. Este tema de investigación es de relevancia para la solución de problemas de gran complejidad matemática, ya que permite que diferentes métodos trabajen en forma cooperativa en la solución de un problema, tomando provecho de las capacidades de cada uno de ellos y del procesamiento paralelo a fin de que los métodos de solución puedan ser ejecutados en diferentes procesadores. Para verificar la eficiencia de la metodología propuesta se utilizan librerías del problema de la literatura especializada obteniendo resultados de buena calidad.**

***Palabras clave*— Problema del cartero viajante, recocido simulado, equipos asíncronos, procesamiento paralelo.**

***Abstract*— In this artice is presented a methodology based on A-Teams to solve the travel salesman problem. In order to implement the solution method are used as agents different versions of the Simulated Annealing. In the approach of the problem the objective funtion is considered the cost of the cycle. This research topic is relevant for the solution of problems with high mathematical complexity, because it permits that different methods work on a cooperative way in the solution of a problem, taking advantage of the capabilities of each one of them and also of the parallel computing aiming to execute the methods using multiple processors. In order to verify the efficiency of the proposed methodology are used libraries of the specialized literature, obtaining results of good quality.**

***Key Word* — Traveling salesman problem, simulated annealing, asynchronous teams, parallel computing.**

I. INTRODUCCIÓN

El problema del agente viajero (PAV) se remonta a la Europa del siglo XVII, donde era común contar con un agente viajero que llevaba un catálogo con productos para atender la demanda de ciudades, pueblos y aldeas alejadas de las ciudades principales. En 1800 el matemático irlandés William Rowand Hamilton diseñó un juego para dos competidores, el juego consiste en hacer un recorrido por 20 puntos de un icosaedro usando las conexiones de la figura e iniciando y terminando en el mismo punto.

La conexión entre el juego de Hamilton y el PAV consiste en que en los dos se busca una ruta que inicie y termine en el mismo lugar sin repetir puntos visitados tal como se muestra en la figura 1.

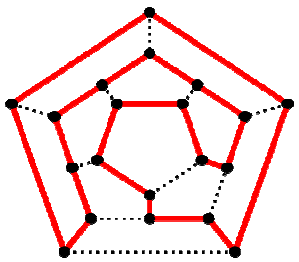

Figura 1. Juego de Hamilton.

[1] Ingeniero de Sistemas y Computación,

[2] Ingeniero de Sistemas y Computación, MSc.

[3] Ingeniero Electricista, PhD.

Fecha de Recepción: 26 de Agosto de 2011
Fecha de Aceptación: 13 de Diciembre de 2011

El 5 de febrero de 1930 en un coloquio en Viena el matemático austríaco Karl Menger plantea por primera vez el PAV. La complejidad del problema radica en que se tienen n ciudades a visitar y que entre cada par de pueblos existe camino directo, pero si en realidad no hay camino directo se construye una distancia infinita, lo que determina que el número de ciclos hamiltonianos diferentes que tiene un mapa de *n* ciudades corresponde a *n!*.

Este problema es uno de los más estudiados en la optimización combinatorial NP-dura [1], [2]. En la literatura existe un gran número de aproximaciones para resolverlo tanto con técnicas exactas como con técnicas heurísticas y metaheurísticas. Dentro de los métodos exactos se incluyen los planos de corte, la relajación de problemas lineales [3], ramificación y acotamiento [4] ramificación y corte [5], [6], [7] y programación dinámica [8], [9], eficientes cuando el problema es de tamaño pequeño.

En el momento de abordar los problemas de gran tamaño han sido utilizadas heurísticas, métodos probabilísticos como los tratados en [1], [10], [11], cadenas de Markov [12], metaheurústicas como Búsqueda Tabú [10], [13], Redes Neuronales [14], Recocido Simulado y Algoritmos Genéticos. [15], [16], [17], [18], [19], [20] y [21].

En 1994 Bixby, Cook y Applegate se reunieron en Rice University en Houston, ellos desarollaron un paquete computacional llamado Cplex, los programas fueron escritos en C++ implementando algoritmos de programación lineal usando puntos interiores, técnicas de podamiento, planos de corte y heurísticas, con el fin de vender el programa desarrollaron instancias para el PAV que denominaron Concorde, este trabajo les hizo merecedores del premio Lanchester que otorgó INFORMS [22].

En este artículo se propone resolver el problema del agente viajero simétrico usando procesamiento paralelo aplicando la arquitectura de equipos asíncronos (A-Teams) empleando agentes de optimización basados en la técnica de recocido simulado. Para comprobar la eficiencia de la metodología propuesta fueron utilizados casos de estudio de la literatura especializada, obteniéndose resultados de buena calidad.

La estructura de este documento es la siguiente: descripción del problema, descripción de la arquitectura de A-Teams, adaptación de la técnica de optimización, Recocido simulado (RS), análisis de resultados y conclusiones.

II. PROBLEMA DEL AGENTE VIAJERO

Existe un conjunto de *n* ciudades, *V = {1, 2, 3, . . ., n}*, y un conjunto de caminos (arcos) uniendo cada una de las ciudades, $(i, j) \in A$. C_{ij} es la distancia para ir de la ciudad *i* a la ciudad *j* donde $(C_{ij} = C_{ji})$ en el caso simétrico. Un agente viajero debe realizar un recorrido partiendo de una ciudad de origen, pasando por cada una de las ciudades una sola vez, y regresando a la ciudad de origen. Se desea encontrar el recorrido de distancia mínima.

Este problema es de compleja formulación matemática, la propuesta por Dantzig, Fulkerson y Johnson [4] es una de las más aceptadas por la comunidad académica y consiste en :

Sea x_{ij} la variable de decisión del problema

$$x_{ij} = \begin{cases} 1 \;\; Si\ el\ arco\ (i, j)\ es\ usado\ para\ hacer\ el\ tour \\ 0 \;\; En\ caso\ contrario \end{cases}$$

Así, el modelo matemático asume la siguiente forma:

$$\min Z(x) = \sum_{(i,j)\in A} c_{ij} x_{ij} \qquad (1)$$

s.a

$$\sum_{\{i:(i,j)\in A\}} x_{ij} = 1 \quad \forall j \in V \qquad (2)$$

$$\sum_{\{j:(i,j)\in A\}} x_{ij} = 1 \quad \forall j \in V \qquad (3)$$

$$\sum_{\{(i,j)\in A:\ i\in U,\, j\in (V-U)\}} x_{ij} \geq 1 \quad 2 \leq |U| \leq |V| - 2 \qquad (4)$$

La ecuación (1) corresponde al cálculo de la función objetivo.
La restricción (2) indica que se puede llegar a cada ciudad desde una única ciudad anterior.
La restricción (3) indica que desde la ciudad *i* se puede pasar a una única ciudad (de la ciudad *i* se puede salir por un único camino).
La restricción (4) evita que se generen subtours.

Aparentemente es un problema sencillo de resolver, pero si se considera un ejemplo con *n=12* ciudades. El número total de ciclos hamiltonianos diferentes son 12! = 479.001.600. Si se resuelven cinco ciclos hamiltonianos en 1 segundo, se necesitarían 95.800.320 segundos o 1.108.8 días más de 3 años para encontrar la solución óptima.

El problema tiene considerables aplicaciones prácticas, aparte de las más evidentes en áreas de logística de transporte, que cualquier negocio de reparto, pequeño o grande, conoce. Por ejemplo, en robótica, permite resolver problemas de fabricación para minimizar el número de desplazamientos al realizar una serie de perforaciones en una plancha o en un circuito impreso. También puede ser utilizado en ruteamiento de vehículos, manufactura flexible, etc.

Para resolver el problema mencionado se implementó un A-Team con agentes basados en RS. A continuación, se describe la arquitectura de los equipos asíncronos.

III. DESCRIPCIÓN DE LA ARQUITECTURA DE A-TEAMS

En problemas de gran complejidad matemática, donde una sola metodología podría no ser suficiente es necesario integrar diferentes técnicas que ofrezcan fortalezas complementarias para la situación que se intenta solucionar. Uno de los primeros interrogantes que debe solucionar el diseñador del sistema que va a resolver el problema es qué habilidades (fortalezas de la técnica de solución) se van a incluir y cómo dichas técnicas se complementan de tal manera que resulte una metodología de solución de gran capacidad.

En un A-Team las habilidades son empaquetadas como agentes. De manera más específica, un A-Team es un sistema multi-población y multi-agente para resolver problemas de optimización. El proceso de cálculo se da de la siguiente manera:

1) El problema que se va a resolver se descompone en sub-problemas.
2) Una población de soluciones candidatas se mantiene para cada uno de los subproblemas mencionados.
3) Dichas poblaciones son cambiadas de manera iterativa por un conjunto de agentes. Las poblaciones de soluciones candidatas pueden intercambiar individuos. Los procesos de cálculo terminan cuando cesan las mejoras en las poblaciones.

Los agentes en los A-Team son idénticos en dos aspectos. En primer lugar, todos ellos son completamente autónomos (trabajan sin supervisión). En segundo lugar todos tienen el mismo ciclo de trabajo (en cada iteración, cada agente realiza los mismos tres pasos: selecciona una solución de la población; modifica la solución seleccionada y luego inserta la solución modificada a la población). En todo, excepto los dos aspectos mencionados anteriormente, los agentes pueden, y usualmente deberían ser diversos. Por ejemplo, algunos podrían ser programas mientras que otros podrían ser humanos. Algunos podrían iterar rápidamente mientras que otros pueden ser lentos. Algunos podrían realizar pequeñas mejoras mientras que otros podrían realizar cambios radicales e incluso innovadores a las soluciones candidatas.

Ventajas de un A-Team

La arquitectura de los A-Team ofrece tres ventajas clave:

1) Modularidad: En un A-Team, los agentes son autónomos y no hay una dependencia entre ellos para su ejecución o comunicación. Por lo tanto, en el proceso del desarrollo del software, los agentes se pueden construir independientemente. Esto permite que la implementación de la solución pueda realizarse modularmente. Los agentes pueden ser fácilmente agregados o eliminados del sistema en cualquier momento. El análisis de la población de soluciones candidatas normalmente indica qué tipo de agente debe ser agregado para mejorar los resultados. La arquitectura A-Team permite incrementar y hacer mantenimiento con mayor facilidad a sistemas que presenten complejidad.

2) Distribuido: La naturaleza asíncrona de los A-Team los hace apropiados para el procesamiento en paralelo.

3) Robustez: La confiabilidad del sistema es muy importante especialmente cuando la metodología se utiliza en entornos de producción. Los A-Teams son robustos porque la falla de un agente no genera la caída de todo el sistema. Si bajo circunstancias particulares un agente produce una solución inválida, o falla mientras genera alguna solución, esto no afectará el funcionamiento de los demás agentes.

Diseñando un sistema orientado a la solución de un problema hay cuatro aspectos de interés que pueden entrar en conflicto:

- Diseño y montaje: Cuánto esfuerzo se necesita para producir una mejora del sistema.
- Calidad de la solución: Qué tan cerca se encontrarán las soluciones obtenidas de la solución óptima.
- Velocidad de solución: Qué tan rápido el sistema completará sus cálculos.
- Robustez: Cómo la calidad de la solución y la velocidad serán afectadas por las fallas del sistema.

Para ver mas detalles de A-Teams [23].

IV. ADAPTACIÓN DE LA TÉCNICA DE OPTIMIZACIÓN

En este trabajo fue utilizada la técnica de optimización de recocido simulado adaptada al problema del cartero viajante mediante una codificación tipo vector, en la cual las posiciones representan el orden en el que deben ser recorridas las ciudades por el cartero y los valores contenidos en el vector codifican las ciudades.

El recocido simulado (RS) es llamado así debido a su analogía con el proceso físico de recocido de sólidos, en el cual un sólido es llevado a temperaturas altas y luego a través de un programa de enfriamiento lento es llevado a una estructura cristalina perfecta (es decir, su estado mínimo de energía), y así se genera un cristal libre de defectos. Si el programa de enfriamiento es lo suficientemente estable, la estructura final será la de un sólido con integridad estructural superior. El algoritmo de recocido simulado establece la conexión entre este tipo comportamiento termodinámico y la búsqueda de un óptimo global en un proceso de optimización.

El RS inicia el proceso con un estado inicial, un cambio aleatorio es propuesto para este estado y un cambio de energía es

generado, ΔE, es calculado. Si el nuevo estado tiene un nivel más bajo de energía que el estado anterior, $\Delta E \leq 0$, el nuevo estado es tomado para la siguiente iteración. Sin embargo, si el nuevo estado tiene un nivel de energía más alto que el anterior, este es aceptado con una probabilidad que depende de un parámetro temperatura, la cual es normalmente disminuida con cada iteración del algoritmo. En la Tabla 1 se muestran los pasos del algoritmo recocido simulado.

Recocido Simulado
1. Escoja una solución inicial *S* 2. Halle la temperatura inicial $T = T_0 > 0$ 3. Seleccione un programa de repetición M, que define el número de iteraciones en cada temperatura *T* 4. Repita: 4.1 Para $m = 0$ hasta *M* haga: a. Escoja una nueva solución S' con el mecanismo de transición propio del agente que encapsula el recocido simulado. b. Sea $\Delta E = E(S)' - E(S)$, donde *E(S)* es la energía de la solución *S* c. Si $\Delta E \leq 0$, acepte la nueva solución, haga $S = S'$ d. De lo contrario, Si $exp(-\Delta E/T) \geq rand(0,1)$, acepte la nueva solución e. De lo contrario rechace la solución 4.2. Reduzca la temperatura de acuerdo al enfriamiento programado 4.3. Incremento de la cadena 5. Hasta la condición de parada.

Tabla 1. Recocido simulado presente en los agentes.

A diferencia del proceso original donde son realizados cambios aleatorios en el proceso de optimización, estas modificaciones son efectuadas a través de mecanismos de transición especializados. Para el problema del cartero viajante diferentes mecanismos han sido propuestos. En este trabajo son tenidos en cuenta los mecanismos de inserción, troca simple, 2-opt y 3-opt.

Cada uno de los agentes cuenta con un mecanismo de transición diferente, como se indica en la Tabla 1 en el paso 4.1.a. cada agente cuenta con una vecindad diferente, por tanto explorarán espacios de soluciones diferentes ofreciendo diversidad a las soluciones que se irán generando en el proceso conjunto de optimización de la metodología propuesta.

En la Figura 2 se muestran las diferentes transiciones que contienen cada uno de los agentes cuando están trabajando colaborativamente.

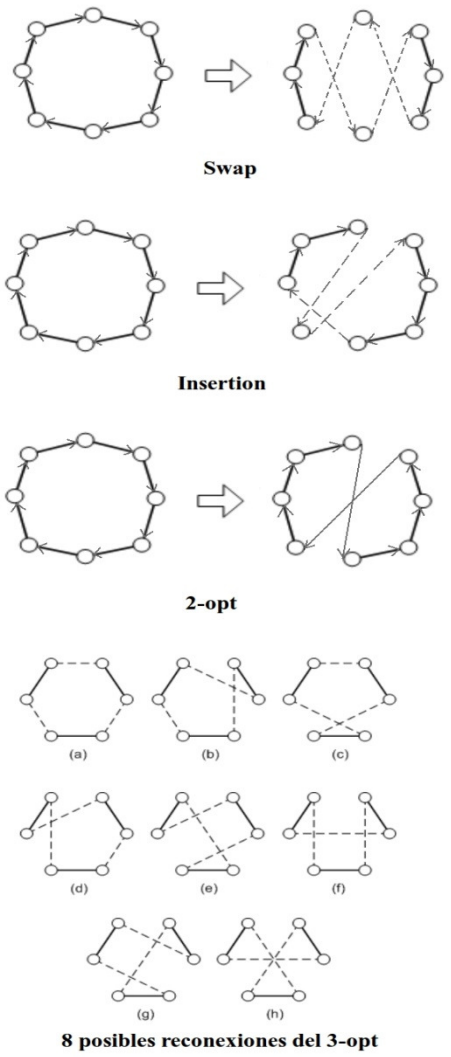

Figura 2. Transiciones de cada uno de los agentes.

En este trabajo, los agentes tienen en común el uso de recocido simulado como metaheurística de optimización. Al encapsularla en agentes se adquiere la robustez de la arquitectura, replicar la metodología con variantes facilita la aplicación de la técnica y la diversidad es generada con mecanismos de transición heterogéneos que guían a los agentes hacia espacios de solución diferentes.

En la Figura 3 se presenta el modelo de A-Team utilizado, este fue implementado en lenguaje C/C++ con librerías de paso de mensajes (MPI) y generación de hilos (OpenMP), distribuido en un clúster con 4 unidades de procesamiento de 2,66 GHz. Para corroborar la eficiencia de la metodología propuesta fueron seleccionados 17 casos de prueba de la literatura especializada.

V. ANÁLISIS DE RESULTADOS.

La metodología fue evaluada con 17 casos de prueba obtenidos de la literatura especializada (disponibles en [24]). Fueron seleccionadas instancias de diferentes dimensiones para observar

el comportamiento del algoritmo y los resultados fueron satisfactorios dado que fue posible alcanzar el óptimo reportado para cada uno de los casos seleccionados.

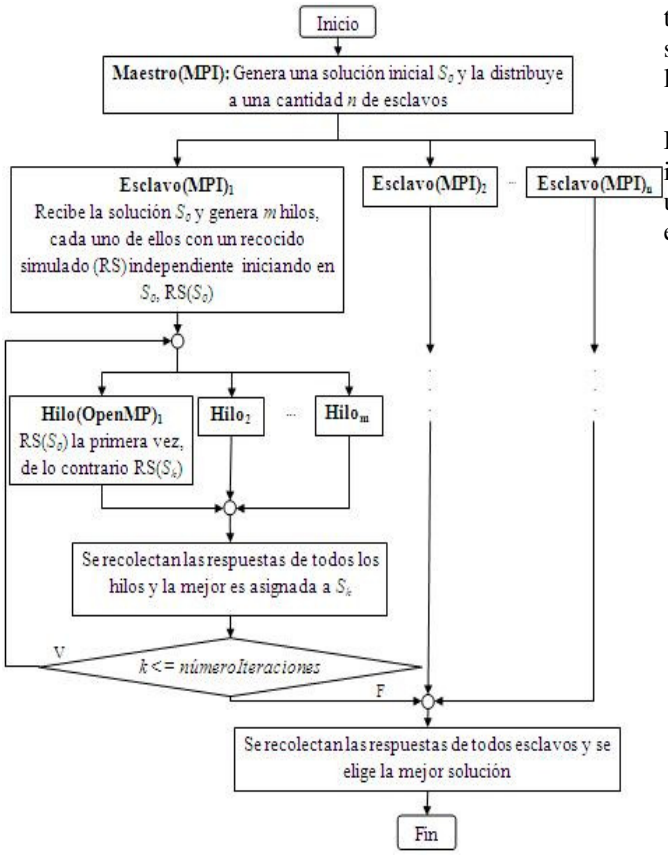

Figura 3. Recocido simulado paralelizado en un clúster empleando MPI y OpenMP.

Los parámetros de la metodología implementada se presentan en la Tabla 2, estos valores fueron obtenidos a través de una búsqueda exhaustiva experimental. El cálculo de la temperatura inicial, que es crucial para el funcionamiento correcto de cualquier recocido simulado, fue realizado realizando un muestreo aleatorio controlado del espacio de soluciones.

Parámetros	
Tamaño inicial de la cadena	3 veces el tamaño del problema
Incremento de la cadena	10%
Tasa de enfriamiento	7%
Criterio de parada	100 iteraciones
Número de hilos en cada esclavo (OpenMP)	5
Número de iteraciones para los hilos	4
Número de esclavos (MPI)	3

Tabla 2. Parámetros y sus valores óptimos del recocido simulado.

Por otro lado, se efectuaron pruebas del algoritmo encapsulado en los agentes, es decir, ejecutándolo en un entorno secuencial y aislado para medir el impacto producido por la arquitectura distribuída con la que cuenta la metodología presentada en este trabajo. Al realizar este enfrentamiento se puede observar que no sólo los tiempos fueron mejorados, sino que también la calidad de las respuestas se incrementa aplicando A-Teams.

De las respuestas obtenidas por los agentes ejecutados individualmente fue escogida la mejor respuesta arrojada por cada uno con su respectivo tiempo de cómputo. En la Tabla 3 se ilustra el resumen del comportamiento de los agentes en *stand-alone*.

Instancia	**Unidades por encima del óptimo reportado**	**Tiempo de Cómputo (Segundos)**
gr24	0	40,0183
fri26	0	43,5457
bayg29	0	48,3522
bays29	0	47,3053
att48	20	84,4737
gr48	0	84,5103
eil51	0	87,2256
berlin52	0	89,4018
st70	0	129,671
eil76	1	152,074
pr76	0	150,005
kroA100	190	208,391
rd100	77	211,806
eil101	7	215,785
lin105	176	223,37
ch150	135	361,658
a280	207	837,952

Tabla 3. Respuesta y tiempo de cómputo de los agentes en un entorno *stand-alone*.

Los resultados arrojados entonces por la metodología que se presenta en este trabajo se muestran en la Tabla 4.

Instancia	**Unidades por encima del óptimo reportado**	**Tiempo de Cómputo (Segundos)**
gr24	0	1,68908
fri26	0	1,87476
bayg29	0	2,07198
bays29	0	2,13237
att48	0	3,66229
gr48	0	3,64438
eil51	0	3,85392
berlin52	0	5,48205
st70	0	7,87146
eil76	0	8,46584

pr76	0	8,36903
kroA100	0	12,3643
rd100	0	12,7972
eil101	0	12,2255
lin105	0	21,8851
ch150	0	24,1884
a280	0	55,3931

Tabla 4. Comportamiento de la metodología propuesta, A-Teams en un entorno distribuido.

VI. CONCLUSIONES

Fue desarrollada una metodología que usa una arquitectura de A-teams en la solución del problema del agente viajero. La arquitectura de A-Teams logró alcanzar todos los óptimos reportados de las instancias elegidas de la literatura especializada, probando su buen funcionamiento para casos de diferentes dimensiones.

La distribución del algoritmo en múltiples procesadores mejoró los tiempos de cómputo respecto a la metodología *stand-alone*.

La simultaneidad de variantes del recocido simulado ofrece la posibilidad de explorar un espacio de soluciones mucho mayor, aumentando las probabilidades de la metaheurística de encontrar respuestas de buena calidad.

En la etapa de ejecución del algoritmo no se presentaron fallos, corroborando la robustez que en teoría deben tener los A-Teams.

REFERENCIAS

[1]. S. Lin and B.W. Kernighan, An effective heuristic algorithm for the traveling salesman problem, Oper. Res. 21 (1973), pp. 498–516.

[2]. E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys, The Traveling Salesman Problem: G.E. Re Guided Tour of Combinatorial Optimization, Wiley and Sons, New York (1985).

[3]. B. Dantzig, D.R. Fulkerson and S.M. Johnson, Solution of a large-scale traveling-salesman problem, Oper. Res. 2 (1954), pp. 393–410.

[4]. M. Padberg and G. Rinaldi, Optimization of a 532-city symmetric traveling salesman problem by branch and cut, Oper. Res. 6 (1987), pp. 1–7.

[5]. H. Crowder and M.W. Padberg, Solving large-scale symmetric traveling salesman problems to optimality, Manage. Sci. **26** (1980), pp. 495–509.

[6]. M.W. Padberg and S. Hong, On the symmetric traveling salesman problem: a computational study, Math. Prog. Stud. **12** (1980), pp. 78–107.

[7]. M. Grötschel and O. Holland, Solution of large-scale symmetric traveling salesman problems, Math. Program. **51** (1991), pp. 141–202.

[8]. B.E. Bellman, Dynamic programming treatment of the traveling salesman problem, J. Assoc. Comput. Mach. **9** (1963).

[9]. Ö. Ergan and J.B. Orlin, A dynamic programming methodology in very large scale neighbourhood search applied to the traveling salesman problem, Discrete Optim. **3** (2006), pp. 78–85.

[10]. D.S. Johnson, More approaches to the traveling salesman guide, Nature **330** (1987), p. 525.

[11]. G.A. Croes, A method for solving traveling salesman problems, Oper. Res. **6** (1958), pp. 791–812. Full Text via CrossRef

[12]. O. Martin, S.W. Otto and E.W. Felten, Large step Markov chains for the traveling salesman problem, Complex Systems **2** (1991), pp. 299–326.

[13]. C.-N. Fiechter, A parallel Tabu search algorithm for large scale traveling salesman problems, Working Paper 90/1, Départment de Mathématiques, École Polytechnique, Fédérale de Lausanne, 1990.

[14]. J.J. Hopfield and D.W. Tank, Neural computation of decisions in optimization problems, Biol. Cybernet. **52** (1985).

[15]. S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, Optimization by simulated annealing, Science **220** (1983), pp. 671–680. View Record in Scopus | Cited By in Scopus (10353)

[16]. E. Bonomi and J.-L. Lutton, The N-city traveling salesman problem: statistical mechanics and the metropolis algorithm, SIAM Rev. **26** (1984), pp. 551–568. Full Text via CrossRef | View Record in Scopus | Cited By in Scopus (45)

[17]. S. Kirkpatrick and G. Toulouse, Configuration space analysis of traveling salesman problems, J. Phys. **46** (1985), pp. 1277–1292. Full Text via CrossRef | View Record in Scopus | Cited By in Scopus (50)

[18]. B.L. Golden and C.C. Skiscim, Using simulated annealing to solve routing and location problems, Naval Res. Logist. Quart. **33** (1986), pp. 261–280.

[19]. J. Lam, An efficient simulated annealing schedule, Ph.D. Dissertation, Department of Computer Science, Yale University, 1988.

[20]. S. Nahar, S. Sahni and E. Shragowitz, Simulated annealing and combinatorial optimization, Int. J. Comp. Aided VLSI Des. **1** (1989), pp. 1–23.

[21]. C.C. Lo and C.C. Hus, An annealing framework with learning memory, IEEE Trans. Syst. Man Cybern. A **28** (1998), pp. 1–13.

[22]. Applegate,D, Bixby R.,Cook W. The traveling salesman problem: A computational study.Princenton University Press,2007

[23]. G. A. Kochenberger. *Handbook of Metaheuristics*. Boston/Dordrecht/London, Kluwer Academic Publishers, University of Colorado at Denver, 2003.

[24]. G. Reinelt. *Library instances,* Universität Heidelberg, Available [online]: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/