A Dual Bi-directional Heuristic Development Framework

Stanley Murairwa[δ]

Abstract

The research designs a multi-start heuristic framework. The heuristic development approach increases the speed of convergence of heuristics to high quality optimum solutions. The multi-start heuristic named the dual bi-directional (DBD) heuristic searches for the global optimum solution in four concurrent directions with a pair search starting from both the beginning and ending nodes. The search terminates when the four optimum tours connect to form an optimum round tour of all the search space nodes. Then, the DBD heuristic starts to improve the found optimum round tour in a unidirectional approach using a global search metaheuristic. The multi-start heuristic framework decreases the non-convergence of the bi-directional approach by introducing the unidirectional heuristic to improve the multi-start heuristic constructed optimum round tour. The development approach will allow the convergence of the bi-directional heuristic.

Keywords: Dual bi-directional heuristic, Heuristic, Global Optimum Solution (GOS), bi-directional heuristic, Optimum solution, Multi-start heuristic

1.0 Introduction

The research proposes a multi-start search framework named the Dual Bi-Directional (DBD) heuristic framework for NP-complete symmetric travelling salesperson problems. The development is a modification of the Greedy Randomised Adaptive Search Procedure (GRASP) by Hart and Shogan (1987) and A* algorithm by Hart *et al.* (1968). The multi-start search development framework also considered the features of the modifications by de Champeaux and Sint (1977), Kaindl *et al*. (1999) and Toptsis *et al*. (2009). The article's main ideas are to reduce the complexity of the NP-complete real life problems and improve the convergence rate and quality of the feasible solutions by implementing a four concurrent search technique. The research by Toptsis *et al*. (2009) provides evidence that the DBD heuristic is faster than the bi-directional algorithm. While the effort of this article is limited to the dual bi-directional procedure as the name implies, the development concept can be extended to any Multi-start heuristic. Therefore, the objective of this research is to develop a dual bi-directional heuristic framework that is capable of increasing the speed of convergence of heuristics to high quality optimum solutions (if not the global optimum solution) within few searching iterations.

2.0 Literature review

The heuristic search algorithms can be classified into two types, namely, unidirectional and bi-directional types (de Chapeaux and Sint, 1977; Toptsis *et al*., 2009). In the unidirectional type (for example Hart *et al*., 1968) there is only one-direction type of process emanating from the source node A and seeking the goal node Z. On the other hand, the bi-directional type (for example Pohl, 1971; de Chapeaux, 1983; Nelson and Toptsis, 1991; 1992) incorporates two types of processes; one forward type search process from A to Z and one reverse type search

[δ] Africa University, Faculty of Management and Administration, Box 1320, Mutare, Zimbabwe
Email: murairwas@africau.edu

process from Z to A. The best-known bi-directional algorithms in Operations Research are by Dijkstra (1959), Bellman (1958) and Ford (1956) while in Artificial Intelligence, it is the A* algorithm by Hart *et al*. (1968). However, Pijls and Post (2008) wrote that the most efficient algorithm is the bi-directional A* which utilises a balanced heuristic as demonstrated by Ikedah *et al*. (1994), Goldberg and Harrelson (2005) and Klunder and Post (2006).

The DBD heuristic is a modification of the A* algorithm by Hart *et al*. (1968) and its refinements by Dechter and Pearl (1985) and de Champeaux and Sint (1977). Pijls and Post (2008) proposed an Algorithm 1 (which is akin to the A* algorithm) with the code running simultaneously on both sides. Toptsis *et al*. (2009) suggested a guided parallel bi-directional algorithm which the researchers proved to be converging faster than the A* algorithm of Hart *et al*. (1968). de Champeaux and Sint (1977) refined the bi-directional search of Pohl (1969; 1971) and produced an improved bi-directional heuristic search algorithm. Jensen *et al*. (2002) proposed a combination of the A* (Hart *et al*., 1968) and Binary Decision Diagram (Bryant, 1986) and produced a Set A* algorithm. Hansen and Zhou (2007) demonstrated how to convert the A* algorithm to find a sequence of improved solutions and converged to the optimum solution. The concept by Hansen and Zhou (2007) was first proposed by Harris (1974) with a bandwidth heuristic search that is related to the weighted heuristic search.

The Perimeter Search algorithm (also known as the bi-directional search) by Dillenburg and Nelson (1993), unlike other bi-directional search algorithms, was developed to avoid some of the pitfalls of attempting to do the two searches simultaneously. Dillenburg and Nelson (1993) stated that the initial best-case scenario for the bi-directional heuristic path algorithm (BHPA) (Pohl, 1971) was for the two searches to meet in the middle. This would reduce an O(b^d) search into O($b^{\frac{d}{2}}$) searches (Dillenburg and Nelson, 1993). However, the two searches do not meet at the midpoint. The meet at the midpoint was eliminated in an improved bi-directional heuristic search by de Champeaux and Sint (1977). Politowski and Pohl (1984) proposed a search switching bi-directional search until the two tours meet. Dillenburg and Nelson (1993) stated that a recent development in bi-directional search algorithms was the BS* algorithm (Kwa, 1989) derived from Pohl's BHPA algorithm with refinements added to eliminate excessive node expansions. Goldberg and Harrelson (2005) provide a list of proposed refinements of the A* algorithm. Holte (2009) noted that the A* algorithm does not always converge to an optimum after meeting but can do so if it continues to search in order to find the optimum solution. Dai & Goldsmith (2007) discovered that the bi-directional heuristic search algorithms outperform unidirectional heuristic search algorithms. Kaindl *et al*. (1999) believes that the traditional heuristics run out of space during the search even for problem instances of moderate size.

2.1 Research gap

There are efforts by researchers and practitioners such as Toptsis *et al*. (2009) and Hansen and Zhou (2007) to improve the speed of heuristics' convergence to high quality optimum solutions if not the global optimum solution within a few searching iterations. Dai and Goldsmith (2007) discovered that the bi-directional heuristics were faster than the unidirectional heuristics. The traditional heuristics get chocked during the search even for moderate problem instances (Kaindl *et al*., 1999). The researchers are trying to push heuristics to find the global optimum solution within a few searching iterations. The question that arises is "How can the bi-directional framework be improved to increase the speed of heuristics' convergence to high quality optimum solutions (if not the global optimum solution) within a few searching iterations?"

3.0 The DBD heuristic

3.1 DBD heuristic mathematical notation

This section presents the mathematical notation used to develop the DBD heuristic framework. Suppose a Travelling Salesperson Problem (TSP) instance of size n is considered, the following variables are defined:

- n is the sample size of the TSP instance;
- ξ is the search space;
- $a\ (a_1, a_2)$ is the start node with a_1 and a_2 starting points;
- $z\ (z_1, z_2)$ is the end node with z_1 and z_2 starting points;
- v, x, y and w are some of the nodes between a and z nodes;
- d_1 is the collection of nodes reached from a to z with known $f_{d_1}(a, z)$ value;
- d_2 is the collection of nodes reached from z to a with known $f_{d_2}(z, a)$ value;
- A_1 is the collection of nodes reached so far from a_1 towards w with known f_{a_1} value;
- A_2 is the collection of nodes reached so far from a_2 towards v with known f_{a_2} value;
- Z_1 is the collection of nodes reached so far from z_1 towards w with known f_{z_1} value;
- Z_2 is the collection of nodes reached so far from z_2 towards v with known f_{z_2} value;
- $\bar{A}_1$ is the collection of nodes not in A_1 but can minimise the tour in A_1;
- $\bar{A}_2$ is the collection of nodes not in A_2 but can minimise the tour in A_2;
- $\bar{Z}_1$ is the collection of nodes not in Z_1 but can minimise the tour in Z_1;
- $\bar{Z}_2$ is the collection of nodes not in Z_2 but can minimise the tour in Z_2;
- $d(x, y)$ is the minimum distance between node x and node y;
- $d(x, y)$ is an estimator of the distance between x and y with $d(x, y) = d(y, x)$;
- $d_{a_1}(y)$ is the minimum distance between a_1 and y for $y \in A_1 \cup \bar{A}_1$ with path in $A_1 \cup \bar{A}_1$;
- $d_{a_2}(y)$ is the minimum distance between a_2 and y for $y \in A_2 \cup \bar{A}_2$ with path in $A_2 \cup \bar{A}_2$;
- $d_{z_1}(y)$ is the minimum distance between z_1 and y for $y \in Z_1 \cup \bar{Z}_1$ with path in $Z_1 \cup \bar{Z}_1$;
- $d_{z_2}(y)$ is the minimum distance between z_1 and y for $y \in Z_2 \cup \bar{Z}_z$ with path in $Z_2 \cup \bar{Z}_2$;
- $D_{a_1}(w) = \min\limits_{y \in \bar{Z}_1}\{d(w, y) + d_{z_1}(y)\}$;
- $D_{a_2}(v) = \min\limits_{y \in \bar{Z}_2}\{d(v, y) + d_{z_2}(y)\}$;
- $D_{z_1}(w) = \min\limits_{y \in \bar{A}_1}\{d(w, y) + d_{a_1}(y)\}$;
- $D_{z_2}(v) = \min\limits_{y \in \bar{A}_2}\{d(v, y) + d_{a_2}(y)\}$;
- $f_{a_1}(x) = d_{a_1}(x) + D_{a_1}(x)$;
- $f_{a_2}(x) = d_{a_2}(x) + D_{a_2}(x)$;
- $f_{z_1}(x) = d_{z_1}(x) + D_{z_1}(x)$;
- $f_{z_2}(x) = d_{z_2}(x) + D_{z_2}(x)$;
- $\gamma(x)$is the finite set of nodes obtained by applicable operators on x;
- $\gamma^{-1}(x)$ is like $\gamma(x)$ but with inverse operators instead and
- $l(j, x)$ is the edge length between j and x.

3.2 Basic concepts

The DBD heuristic, which is in two phases, increases the chances of converging to the global optimal solution through its transition from dual bi-directional to unidirectional approach. The two phases just like the GRASP (Hart and Shogan, 1987) are the construction and improvement phases. The latter can be done by a local search heuristic such as the λ-Opt procedure {2-Opt (Flood, 1956; Croes, 1958) or 3-Opt (Bock, 1958)} or a global search heuristic such as Genetic Algorithm (Bagley, 1967; Holland, 1975), Tabu Search (Glover, 1990; 1989), Ant Colony (Dorigo, Maniezzo and Colorni, 1991) or Simulated Annealing (Metropolis, Rosenbluth, Rosenbluth, Teller and Teller, 1958). However, the construction phase is the dual bi-directional approach while the improvement phase is the unidirectional approach. Unlike the other bi-directional search algorithms, the DBD heuristic is executed in four (a_1, a_2, z_1, z_2) concurrent searches. As described by Toptsis *et al.* (2009) in parallel bi-directional algorithm, the DBD heuristic divides the search space into four areas and generates four optimum tours that are connected together to form an optimum round tour. The optimum round tour is then improved by the unidirectional algorithm in the improvement phase to enhance its quality.

The DBD heuristic locates the starting node a (with a_1 and a_2 starting points) and ending node z (with z_1 and z_2 starting points) in the search space. The a and z should be the beginning and ending extreme nodes in the search space. The search space is divided into four searchable areas. Suppose w and v are the meeting nodes in the search space as presented in Figure 1. The two meeting nodes should not necessarily be at the centre of the search space. Then, the two forward tours are (a_1, w) and (a_2, v) while the backward tours are (z_1, w) and (z_2, v). The four optimum tours are constructed simultaneously and this means that the algorithm runs concurrently on both nodes. The DBD heuristic concept is depicted in Figure 1.

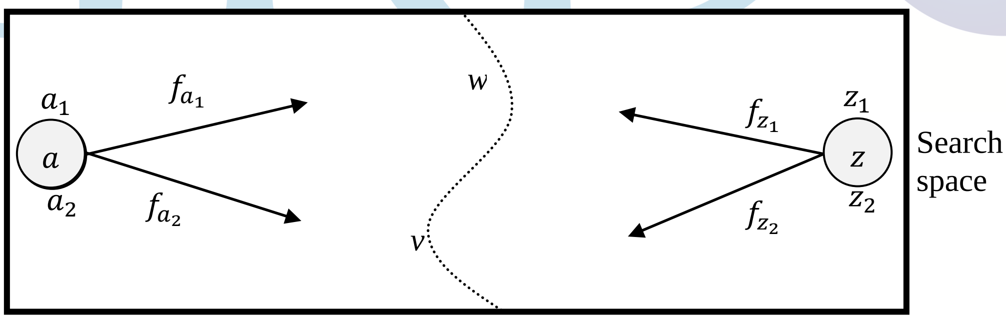

Figure 1: The DBD heuristic concept

Figure 1 shows that the DBD heuristic starts searching concurrently in four directions towards the centre with a pair search starting at both the beginning and ending nodes. Thus, the minimum distance function is given by:

$$f_d(\tau) = f_{d_1}(a, z) + f_{d_2}(z, a), \tag{1}$$

where $f_{d_1}(a, z) = f_{a_1}(a_1, w) + f_{z_1}(z_1, w)$ and $f_{d_2}(z, a) = f_{z_2}(z_2, v) + f_{a_2}(a_2, v)$. The addition of the j^{th} node to a_1 or a_2 or z_1 or z_2 from the search space ξ is determined by the evaluation function:

$$\underset{j\in\xi}{\text{Min}}\left\{f_{a_1}(a_1,j);\ f_{a_2}(a_2,j);\ f_{z_1}(z_1,j);\ f_{z_2}(z_2,j)\right\}, \tag{2}$$

The ties are broken by considering the next minimising node in the search space. The DBD algorithm terminates the construction of the tour when the four optimum tours meet at w and v to form an optimum round tour. Instantly, the unidirectional algorithm starts to improve the found optimum round tour in a unidirectional approach. The research recommends that the global search metaheuristic be employed to improve the found optimum round tour in order to avoid the challenges that are associated with most local search heuristics. Kaindl *et al*. (1999) affirmed that switching from bi-directional to unidirectional helps to prove the optimality of the solutions found. The unidirectional algorithm terminates the improvement of the round tour when the termination criterion is satisfied.

3.3 DBD heuristic conditions

The following conditions apply for the DBD heuristic:

a) Each of the starting nodes should:
   i) have memory to store search signals and
   ii) clearly be marked for identification.
b) The coordinating point (tabu list) stores the visited nodes.
c) The DBD heuristic terminates twice during the search for the global optimum solution. The termination conditions are:
   i) The DBD algorithm terminates the construction of the round tour when the four optimum tours meet at v and w in the search space to form an optimum round tour with all the nodes of the search space connected.
   ii) The unidirectional algorithm terminates the improvement of the optimum round tour when it can no longer be improved in a fixed number of consecutive iterations.

3.4 DBD heuristic general framework

Step 1: Defining the search space: The DBD algorithm detects all the TSP instance nodes in the search space and identifies the initial node (τ_0) and end node (τ_n). It should be noted that (τ_0) and (τ_n) are the two extreme nodes a and z respectively in the search space.

Step 2: Allocation of two paired makers (tokens): each of the two nodes (a, z) is allocated two different tokens.

Step 3: Initialisation: each node releases the two different tokens simultaneously. The tours are constructed towards the centre of the search space by a heuristic procedure. For $R = 1$, the algorithm generates the initial solution through random or construction. For $R > 1$, the algorithm retains the immediate past run tours as the starting tours if the tour generated is of less quality to the previous run solution.

Step 4: Optimum round tour improvement: the unidirectional algorithm is applied to improve the quality of the found optimum round tour.

Step 5: Terminating criteria: the DBD heuristic terminates with an optimum solution when the stopping criterion for the unidirectional algorithm is satisfied.

3.5 DBD Algorithm

1. Identify the starting node $a(a_1, a_2)$ and ending node $z(z_1, z_2)$ and divide the search space into four searchable areas as shown in Figure 1.

2. Put a_1 in $\bar{A}_1$ and z_1 in $\bar{Z}_1$ with $f_{a_1}(a_1) := f_{z_1}(z_1) := d_1(a_1, z_1)$ and a_2 in $\bar{A}_2$ and z_2 in $\bar{Z}_2$ with $f_{a_2}(a_2) := f_{z_2}(z_2) := d_2(a_2, z_2)$. Delete a_1, a_2, z_1 and z_2 from A_1, A_2, Z_1 and Z_2 respectively.
3. If $\bar{A}_1 \cup \bar{A}_2 \cup \bar{Z}_1 \cup \bar{Z}_2 \neq \emptyset$, proceed to step 4; otherwise stop without a solution.

4.1 Select w in $\bar{A}_1$ with $f_{a_1}(w) = \min_{y_{11} \in \bar{A}_1} f_{a_1}(y_{11})$, remove w from $\bar{A}_1$ and put w in A_1, let descendants $(w) := \gamma(w)$.

4.2 Select v in $\bar{A}_2$ with $f_{a_2}(v) = \min_{y_{12} \in \bar{A}_2} f_{a_2}(y_{12})$, remove v from $\bar{A}_2$ and put v in A_2, let descendants $(v) := \gamma(v)$.

4.3 Select w in $\bar{Z}_1$ with $f_{z_1}(w) = \min_{y_{11} \in \bar{Z}_1} f_{z_1}(y_{11})$, remove w from $\bar{Z}_1$ and put w in Z_1, let descendants $(w) := \gamma^{-1}(w)$.

4.4 Select v in $\bar{Z}_2$ with $f_{z_2}(v) = \min_{y_{12} \in \bar{Z}_2} f_{z_2}(y_{12})$, remove v from $\bar{Z}_2$ and put v in Z_2, let descendants $(v) := \gamma^{-1}(v)$.

5. If $w \in \bar{Z}_1$ (in 4.1) and $v \in \bar{Z}_2$ (in 4.2) then stop with an optimum solution:

$$Min\{f_d(\tau_i), f_d(\tau_{i-1})\} = \frac{1}{2}\{[|f_d(\tau_i) + f_d(\tau_{i-1})|] - [|f_d(\tau_i) - f_d(\tau_{i-1})|]\}, \tag{3}$$

with $f_d(\tau_1) = f_d(\tau_0)$ for the first run $i = 1$ as the starting tour to the unidirectional phase.

6. If descendants (w) and descendants (v) = Ø, then return to step 3.

7.1 Let $x \in$ descendants (w) and remove x from $\bar{A}_1$.

7.2 Let $x \in$ descendants (v) and remove x from $\bar{A}_2$.

7.3 Let $x \in$ descendants (w) and remove x from $\bar{Z}_1$.

7.4 Let $x \in$ descendants (v) and remove x from $\bar{Z}_2$.

8.1.1 If $x \in \bar{A}_1$, then {if $f_{a_1}(w) + l(w, x) < f_{a_1}(x)$, then $f_{a_1}(x)$: $f_{a_1}(w) + l(w, x)$; if $f_{a_1}(x) + h_{a_1}(x) < f_{a_1}(x)$; return to step 6}.

8.1.2 If $x \in A_1$, then {if $f_{a_1}(w) + l(w, x) < f_{a_1}(x)$, then $f_{a_1}(x)$: $f_{a_1}(w) + l(w, x)$; if $f_{a_1}(x) + h_{a_1}(x) < f_{a_1}(x)$, then [$f_{a_1}(x) := g_{a_1}(x) + h_{a_1}(x)$: remove x from A_1 and put x in $\bar{A}_1$]; return to step 6}.

8.1.3 Put x with its value $f_{a_1}(x)$ in $\bar{A}_1$, then go to step 6.

8.2.1 If $x \in \bar{A}_2$, then {if $f_{a_2}(w) + l(w, x) < f_{a_2}(x)$, then $f_{a_2}(x)$: $f_{a_2}(w) + l(w, x)$; if $f_{a_2}(x) + h_{a_2}(x) < f_{a_2}(x)$; return to step 7}.

8.2.2 If $x \in A_2$, then {if $f_{a_2}(w) + l(w, x) < f_{a_2}(x)$, then $f_{a_2}(x)$: $f_{a_2}(w) + l(w, x)$; if $f_{a_2}(x) + h_{a_2}(x) < f_{a_2}(x)$, then [$f_{a_2}(x) := g_{a_2}(x) + h_{a_2}(x)$: remove x from A_2 and put x in $\bar{A}_2$]; return to step 6}.

8.2.3 Put x with its value $f_{a_2}(x)$ in $\bar{A}_2$, then go to step 6.

8.3.1 If $x \in \bar{Z}_1$, then {if $f_{z_1}(w) + l(w, x) < f_{z_1}(x)$, then $f_{z_1}(x)$: $f_{z_1}(w) + l(w, x)$; if $f_{z_1}(x) + h_{z_1}(x) < f_{z_1}(x)$; return to step 6}.

8.3.2 If $x \in Z_1$, then {if $f_{z_1}(w) + l(w, x) < f_{z_1}(x)$, then $f_{z_1}(x)$: $f_{z_1}(w) + l(w, x)$; if $f_{z_1}(x) + h_{z_1}(x) < f_{z_1}(x)$, then [$f_{z_1}(x) := g_{z_1}(x) + h_{z_1}(x)$: remove x from Z_1 and put x in $\bar{Z}_1$]; return to step 6}.

8.3.3 Put x with its value $f_{z_1}(x)$ in $\bar{Z}_1$, then go step 6.

8.4.1 If $x \in \bar{Z}_2$, then {if $f_{z_2}(w) + l(w, x) < f_{z_2}(x)$, then $f_{z_2}(x)$: $f_{z_2}(w) + l(w, x)$; if $f_{z_2}(x) + h_{z_2}(x) < f_{z_2}(x)$; return to step 6}.
8.4.2 If $x \in Z_2$, then {if $f_{z_2}(w) + l(w, x) < f_{z_2}(x)$, then $f_{z_2}(x)$: $f_{z_2}(w) + l(w, x)$; if $f_{z_2}(x) + h_{z_2}(x) < f_{z_2}(x)$, then [$f_{z_2}(x)$: = $g_{z_2}(x) + h_{z_2}(x)$: remove *x* from Z_2 and put *x* in $\bar{Z}_2$]; return to step 6}.
8.4.3 Put *x* with its value $f_{z_2}(x)$ in $\bar{Z}_2$, then go to step 6.

The reason for introducing the DBD heuristic development approach is that each of the four searches has complexity $O(b^{\frac{4}{d}})$ (in Big O notation) and $O(b^{\frac{4}{d}} + b^{\frac{4}{d}} + b^{\frac{4}{d}} + b^{\frac{4}{d}})$ and according to Russell and Norvig (2003) is much less than the running time of one search from the beginning to the goal which would be $O(b^d)$. This means a great improvement in the convergence speed of the DBD heuristic provided the improvement algorithm does not take long to terminate.

5.0 Conclusion

The problem of the unidirectional heuristics is the time taken to find an optimum solution of high quality. The article proposed a DBD heuristic development framework that improves the search performance and quality of the optimum solutions. Thus, the heuristic development approach improves the convergence of the heuristic and solves large NP-complete problems within a few iterations. There is need to further improve and implement the DBD heuristic to solve large NP-complete real life TSP instances.

6.0 Areas for further research

The second part of the DBD heuristic development framework will implement it to solve selected real life NP-complete TSP instances.

References

Bagley, J. D. (1967). The behaviour of adaptive systems which employ genetic and correlation algorithms. Ph.D. thesis, University of Michigan.
Bellman, R. (1958). On a routing problem. Quarterly of Applied Mathematics, 16(1), 87-90.
Bock, F. (1958). An Algorithm for Solving Traveling-Salesman and Related Network Optimization Problems. Presented at the 14th National Meeting of the Operations Research Society of America. S t. Louis.
Croes, G. A. (1958). A Method for Solving Travelling-Salesman Problems, Operations Research, 6 (6), 791 – 812.
Dai, P., & Goldsmith, J. (2007). Multi-threaded BLAO* algorithm. In Proc. 20th International FLAIRS Conference, 56–62.
de Champeaux, D. & Sint, L. (1977). An improved bidirectional heuristic search algorithm. J. ACM, 24(2), 177-191.
de Champeaux, D. (1983). Bidirectional heuristic search again. Journal of ACM 30(1), 22 – 32.
Dechter, R. & Pearl, J. (1985). Generalized Best-First Search Strategies and the Optimality of A*. J. ACM, 32(3), 505-536.
Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. Numerische Math-Ematik, 1, 269-271.
Dillenburg, J. F. & Nelson, P. C. (1973). Improving the efficiency of depth first search by cycle elimination. Information processing letters, 45(1), 5 – 10.

Dorigo, M., Maniezzo, V., & Colorni, A. (1991). Positive feedback as a search strategy. Technical Report, n. 91-016, Politecnico di Milano.

Flood, M. M. (1956). The travelling-salesman problem. Operations Research. 4, 61-75.

Ford, L. R. (1956). Network Flow Theory. Technical Report P-923 Rand Corporation, Santa Monica CA.

Glover F. (1989), Tabu Search - Part I, Operations Research Society of America (ORSA) Journal on computing, Vol. 1, No. 3, pp 190-206.

Glover F. (1990), Tabu Search - Part II, Operations Research Society of America (ORSA) Journal on computing, Vol. 2, No. 1, pp 4-32.

Goldberg, A. V. & Harrelson, C. (2005). Computing the Shortest Path: A* Search Meets Graph Theory. 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'05).

Hansen, E. A. & Zhou, R. (2007). Anytime Heuristic Search. Journal of Artificial Intelligence Research, 28, 267-297.

Harris, L. (1974). The heuristic search under conditions of error. Artificial Intelligence, 5 (3), 217 – 234.

Hart, J. P. & Shogan, A.W. (1987). Semi-greedy heuristics: An empirical study. OR Letters, 6, 107-114.

Hart, P., Nilsson, N., & Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. IEEE Trans. Syst. Science and Cybernetics, SSC, 4(2), 100-107.

Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. Ann Arbor, University of Michigan Press.

Holte, R. C. (2009), Common Misconceptions Concerning Heuristic Search, Computing Science Department, University of Alberta Edmonton, Canada.

Ikeda, T. K., Hsu, M., Inai, H., Nishimura, S., Shimoura, H., Hashimoto, T., Tenmoku, K., & Mitoh, K. (1994). A Fast Algorithm for Finding Better Routes by AI Search Techniques. In the proceedings of the Vehicle Navigation and Information Systems Conference, IEEE.

Jensen, R., Bryant, R., & Veloso, M. (2002). Set A*: An efficient BDD-based heuristic search algorithm. In Proceedings of the 18th National Conference on Artificial Intelligence (AAAI-02), 668–673.

Kaindl, H., Kainz, G., Steiner, R., Auer, A., & Radda, K. (1999), Switching from bidirectional to unidirectional search, In Proc, Sixteen International Joint Conference on Artificial Intelligence (IJCAI-99), San Francisco, CA: Morgan Kaufmann Publishers.

Klunder, G. A., & Post, H. N. (2006). The Shortest Path Problem on Large Scale Real Road Net-Works. Networks, 48(4), 182-194.

Kwa, J. B. H. (1989). BS*: An Admissible Bidirectional Staged Heuristic Search Algorithm. Artificial Intelligence, 38(1), 95-109.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M. N., Teller, A.H. and Teller, E. (1958), Equations of State Calculations by Fast Computing Machines, J. Chem. Phys. 21, 1087-1092.

Nelson, P. C., & Toptsis, A. A. (1991). Search Space Clustering in Parallel Bidirectional Heuristic Search. In Proc. Of the 4th UNB Artificial Intelligence Symposium, New Brunswick, Canada in September, 563 – 573.

Nelson, P. C., & Toptsis, A. A. (1992). Unidirectional and Bidirectional Search Algorithms. IEEE Software, 9(2), 77 – 83.

Pijls, W. & Posty, H. (2008). A new bidirectional algorithm for shortest paths. Econometric Institute Report EI 2008-25.

Pohl, I. (1969). Bi-directional and heuristic search in path problems. SLAC-104, UC-32, Stanford university, California.

Pohl, I. (1971). Bi-directional search. In Machine Intelligence. Edinburgh University Press, 6, 127-140.

Politowski, G., & Pohl, I. (1984). D-Node Retargeting in Bidirectional Heuristic Search. In Proc. of the AAAI-84, 274-277.

Russell, S. J. & Norvig, P. (2003). Artificial Intelligence: A Modern Approach (2nd Ed.). Upper Saddle River, New Jersey: Prentice Hall.

Toptsis, A. A., Chaturvedi, R. A., & Feroze, A. (2009). Kohonen-guided Parallel Bidirectional Voronoi-assisted Heuristic Search. International Journal of Advanced Science and Technology, 5, 15 – 34.