# TOWARDS A BETTER UNDERSTANDING OF BIDIRECTIONAL SEARCH

Henry W. Davis
Randy B. Pollack
Thomas Sudkamp

Wright State University

ABSTRACT

Three admissible bidirectional search algorithms have been described in the literature: A Cartesian product approach due to Doran, Pohl's BHPA, and Champeaux and Sint's BHFFA2. This paper describes an algorithm, GP, which contains the latter two and others. New admissibility results are obtained. A first order analysis is made comparing the run times of Cartesian product search, two versions of GP, and unidirectional $A^*$. The goal is to gain insight on when bidirectional search is useful and direction for seeking better bidirectional search algorithms.

## 1. INTRODUCTION

A problem with Pohl's BHPA [7,8] was that search trees did not meet near the middle and the algorithm performed poorly. To remedy this, Champeaux and Sint [2,3] proposed using a "front-to-front" heuristic. Their first algorithm based on this, BHFFA, solves the problem of the search trees meeting in the middle and generally gives a higher "quality" of solution than unidirectional $A^*$. Unfortunately it runs longer and is not admissible. To deal with the admissibility problem Champeaux [1] has described a somewhat complicated algorithm, BHFFA2, which also uses "front-to-front" heuristics.

The algorithm of section 2, GP ("generalized Pohl"), restates Pohl's BHPA with greater generality and adds additional features. One feature (step (2.2)) gives high symmetry to the search. The result is that GP includes BHFFA2, as well as BHPA. For completeness we add a feature which allows GP to be used with graphsearch as well as ordered search procedures. We consider other dynamic heuristics than the one used by Champeaux and Sint. One of them ((2) in section 2.3) makes Pohl's original BHPA admissible while assuring that the search trees meet in the middle, the major goal of BHFFA2. Another (GP2 in section 4) reduces some of the list processing and H-calculation overhead from the traditional OPEN-OPEN approach. In section 3 we show that GP is admissible in a variety of situations not previously considered. We also state a result that one may prune OPEN $\cup$ CLOSED in the latter stages of a GP-search without affecting admissibility.

Section 4 makes a first order analysis of where several algorithms spend their run time. Two versions of GP, Cartesian product search, and unidirectional $A^*$ are compared in several heuristic situations. The number of nodes expanded, the number of H-calculations made, and the amount of list processing are examined in a worst case analysis using a search space previously considered by Pohl, Champeaux, and Sint. The results suggest that, compared to unidirectional search, GP performs favorably with respect to nodes expanded and list processing, but unfavorably with respect to H-calculations unless the heuristic is very weak. The fact that in most of the categories considered (Table 3) some bidirectional algorithm performs better than unidirectional search suggests that substantial improvement in admissible bidirectional search algorithms may be possible.

We have provided a proof of the main admissibility theorem. Due to a shortage of space we do not include proofs for other results mentioned. They will be submitted for publication.

## 2. A GENERALIZED POHL ALGORITHM FOR BIDIRECTIONAL SEARCH

### 2.1 Assumptions and Notation.

Assume that the search space is a locally finite graph, G, whose arc lengths are bounded uniformly above zero. Arcs may be traversed in either direction. We seek a path connecting s, t $\in$ G.

The following notation is used:

| | |
|---|---|
| $x$ | s or t |
| $\bar{x}$ | s if x denotes t and t if x denotes s. |
| x_EXPAND | A node expansion routine described later. |
| x_OPEN | Set of nodes which have been discovered (generated) by x_EXPAND and are awaiting possible expansion. |
| x_CLOSED | Set of nodes which have been x_expanded (and are not currently in x_OPEN in the ordered search case). |
| $H^*(m,n)$ | Actual cost of a least cost path from m to n. $H^*(m,n) = \infty$ if no such path exists. |
| $H(m,n)$ | Heuristic estimate of $H^*(m,n)$. |
| $g_x(m)$ | Cost of least cost path so far found from x to m by GP. |
| $g_x^*(m)$ | Same as $H^*(x,m)$. |
| $h_x^*(m)$ | Same as $H^*(m,\bar{x})$. |
| $h_x(m)$ | A heuristic function which estimates $h_x^*(m)$. |
| $f_x(m)$ | $g_x(m) + h_x(m)$ |

$c(p,m)$ Cost of a least cost arc from p to m in G when such an arc exists. Assume $c(p,m) = c(m,p)$.

$PT_x(m)$ Parent of m with respect to a search tree rooted at x.

AMIN Cost of least cost path between s, t which the algorithm has so far discovered. Initially AMIN $= \infty$.

x_TESTi x_TESTi is one of the sets x_OPEN, x_CLOSED, x_OPEN $\cup$ x_CLOSED, x_CLOSED $\cup$ {x}, i=1,2. These sets are used to determine when to update AMIN. Admissibility results hold for different combinations of x_TESTi values, i=1,2.

## 2.2 GP

We use the following node expansion routine:

x_EXPAND a node $p \in$ x_OPEN
  For each neighbor m of p do
    If $m \notin$ x_OPEN $\cup$ x_CLOSED then
      $PT_x(m) \leftarrow p$
      $g_x(m) \leftarrow g_x(p) + c(p,m)$
      x_OPEN $\leftarrow$ x_OPEN $\cup$ {m}
    If $m \in$ x_OPEN $\cup$ x_CLOSED with $g_x(p) + c(p,m) < g_x(m)$ then
      $g_x(m) \leftarrow g_x(p) + c(p,m)$
      $PT_x(m) \leftarrow p$
      If $m \in$ x_CLOSED then "revitalize m" via ordered search or graphsearch (see below)
    Remove p from x_OPEN
    x_CLOSED $\leftarrow$ x_CLOSED $\cup$ {p}

If ordered search is being used then "revitalize m" means to place m into x_OPEN. In the graphsearch case it means to recursively update the $g_x$ and $PT_x$ values of all of m's G-neighbors and whatever of their G-neighbors that have been so far generated. See [6;2.2.2].

Algorithm GP

(1) x_OPEN $\leftarrow$ {x}, $g_x(x) \leftarrow 0$, $PT_x(x) \leftarrow x$, x=s,t. AMIN $\leftarrow \infty$.

(2) Do steps (2.1) through (2.4) until either (a) or (b) is true:
  (a) x_OPEN $= \emptyset$, x=s or t.
  (b) AMIN $\leq \min \{f_x(y)\colon y \in$ x_OPEN}, x=s or t.

  (2.1) For x=s or t choose some $w \in$ x_OPEN such that $f_x(w) = \min \{f_x(y)\colon y \in$ x_OPEN}. x_EXPAND w. In steps (2.2)-(2.4) x has the value assigned in this step (s or t).

  (2.2) [This step is optional.] If ordered search is being used then any or all of (a), (b), (c) below may be performed. (b),(c) may be performed many times.
    (a) If $w \in \bar{x}$_OPEN then $\bar{x}$_EXPAND w.
    (b) A node n in $\bar{x}$_CLOSED is removed from $\bar{x}$_CLOSED and placed in $\bar{x}$_OPEN provided that (within this same instance of the step 2 do-loop) either $g_{\bar{x}}(n)$ is lowered or n is assigned its first $g_{\bar{x}}$ value.
    (c) Same as (b) with x, $\bar{x}$ interchanged.

  (2.3) For each new member $y \in$ x_TEST1 $\cap$ $\bar{x}$_TEST2 AMIN $\leftarrow$ min(AMIN, $g_s(y) + g_t(y)$).

  (2.4) [This step is only needed if graphsearch is being used] for each $y \in$ x_TEST1 $\cap$ $\bar{x}$_TEST2 such that $g_x(y)$ has been lowered AMIN $\leftarrow$ min(AMIN, $g_s(y) + g_t(y)$).

(3) If AMIN $< \infty$ then report the solution path associated with the current AMIN; else report failure.

## 2.3 Special Cases of GP

To obtain Pohl's BHPA [7,8] set x_TEST1 = x_CLOSED, x_TEST2 = x_CLOSED $\cup$ {x}, omit steps (2.2), (2.4) and use ordered search. The reason x_TEST2 must be x_CLOSED $\cup$ {x} and not x_CLOSED is so that GP works when it is run unidirectionally. Pohl got around this by initially closing both s,t. This technical alteration is the only way GP differs from BHPA when steps (2.2), (2.4) are omitted. Pohl showed that when the $h_x$ are optimistic (ie $h_x \leq h_x^*$) BHPA is admissible. (In [8] consistency is also assumed but [7;page 99], points out that consistency is not required.)

The Champeaux-Sint "front-to-front" heuristic function is given by

(1) $$h_x(z) = \min\{H(z,y) + g_{\bar{x}}(y)\colon y \in \bar{x}\text{\_OPEN}\}.$$

To obtain BHFFA2 from GP one uses (1) and makes appropriately a number of the choices left arbitrary in GP, namely:

(a) In (2.1) choose x=s or t so as to avoid expanding a node from s_OPEN $\cap$ t_OPEN whenever possible.

(b) When in (2.1) GP is forced to expand a node $w \in$ s_OPEN $\cap$ t_OPEN it chooses w to be that node (or one of those nodes) in {y: $y \in$ x_OPEN, $f_x(y)$ is minimum} whose $g_s(w) + g_t(w)$ value is as small as possible.

(c) (2.4) is omitted and ordered search is used.

(d) Champeaux's routines EXPAND1 and EXPAND2 are obtained in GP by performing steps (2.2a), (2.2b), and (2.2c) whenever possible.

(e) Set x_TESTi = x_CLOSED, i=1,2, x=s,t.

It is shown in [1] that if H is optimistic (ie, $H \leq H^*$) then BHFFA2 is admissible. It is not hard to find examples of H being optimistic while the heuristic function $h_x$ based on H (ie (1) holds) is not optimistic. Interestingly there is a simple "front-to-front" heuristic function which is optimistic. Therefore it makes Pohl's BHPA admissible while assuring that the search trees meet in the middle. We show in an appendix that when H is optimistic so is

(2) $$h_x(z) = \begin{cases} \min\{H(z,y) + g_{\bar{x}}(y)\colon y \in \bar{x}\text{\_OPEN}\} & \text{if } z \notin \bar{x}\text{\_CLOSED} \\ \min\{g_{\bar{x}}(z)\} \cup \{H(z,y) + g_{\bar{x}}(y)\colon y \in \bar{x}\text{\_OPEN}\} & \text{otherwise} \end{cases}$$

The definition of equation (1) can be extended as follows:

Let $S(x) \subset$ x_OPEN $\cup$ x_CLOSED, x=s,t. Define

(3) $$h_x^{S,H}(z) = \min\{H(z,y) + g_{\bar{x}}(y)\colon y \in S(\bar{x})\}$$

We call S(x) a target set and say S is admissible if GP has an admissibility theorem for $h_x^{S,H}$ when H is optimistic. This does not imply that $h_x^{S,H}$ is optimistic, although it may be. Pohl and Champeaux, respectively, showed that {x} and x_OPEN are admissible. Another admissible set is $S_1(x) = \{x\} \cup$ x_CLOSED; in fact, for this S, $h_x^{S,H}$ is optimistic (see appendix). An advantage of $S_1(x)$ over x_OPEN is that it's smaller. The problem with such a relatively static non-frontal target set is that H-calculations may become fixed to an interior node (eg, s or t) which erroneously looks close to the opposite front due to an unseen hill. $S_2(x) = \{PT_x(y)$: $y \in$ x_OPEN} is smaller than $S_1(x)$ and doesn't have the interior node problem. It is admissible. One may extend the notion of target set to allow dependence on two variables and, thereby, incorporate (2) into (3). Let

$$S(\bar{x},z) = \begin{cases} \bar{x}\_OPEN \text{ if } z \notin \bar{x}\_CLOSED \\ \{z\} \cup \bar{x}\_OPEN \text{ otherwise} \end{cases}$$

Then S is admissible: Replacing $S(\bar{x})$ in (3) with $S(\bar{x},z)$ gives (2).

## 3. ADMISSIBILITY

### 3.1 Admissibility Theorem.

A search algorithm is admissible if, for any graph in which a solution path exists, it always terminates with a minimal cost path. We make the assumptions stated in section 2.1.

Theorem GP is admissible is either (i) or (ii) hold:

(i) $h_x$ is optimistic and x_TEST2 $\not\subset$ x_CLOSED, x=s, t (Pohl's BHPA).

(ii) $h_x = h_x^{S,H}$ is given by (3) in section 2.3; H is optimistic; S(x) is either x_OPEN or $\{PT_x(y) : y \in$ x_OPEN}; and at least one of (a) or (b) hold:

- (a) Ordered search is used and (2.2a), (2.2b), (2.2c) of GP are executed whenever possible (Champeaux's BHFFA2).
- (b) It is the case that
  - (b1) x_TEST2 $\not\subset$ x_CLOSED, x=s,t, and
  - (b2) either x_TEST1 $\supset$ x_OPEN for x=s,t or x_TEST2 $\supset$ x_OPEN for x = s,t.

The theorem may be proven by technical modifications of the original admissibility arguments in [5], a testimony to the robustness of those arguments. One first proves the classical "partial solution on open" lemma and uses it to eliminate each of the following cases: (1) GP never halts; (2) GP halts with no solution; and (3) GP halts with a non-optimal solution. We prove here GP admissibility only for assumptions (ii). Assumption (i) may be handled along the lines of [7;pp 98 ff].

Lemma Assume (a) there is a path in G connecting s,t; (b) GP has not yet found a minimal cost path; (c) GP has just completed step(1) and zero or more iterations of the outer loop. Then, if (i) or (ii) holds, there are nodes $m(x) \in$ x_OPEN such that $f_x(m(x)) \leq L$, where L is the cost of an optimal path (x=s,t).

Proof We give the proof for (ii) using Champeaux's argument in [1;Lemma 1]. Assume, first, case (iib). Let $\mu = (s=x_0, x_1,\ldots,x_m = t)$ be an optimal path. Not all the $x_p$ are in x_CLOSED because otherwise $\mu$ would have been discovered (x=s,t): this is because (b1) assures that at least $\bar{x} \in$ x_TEST1 $\cap$ $\bar{x}$_TEST2 so steps (2.3), (2.4) would have found $\mu$. Take j least and k greatest such that $x_j \in$ s_OPEN and $x_k \in$ t_OPEN. $j \leq k$ because otherwise, by (b2), $\mu$ would have been discovered. Since $x_l$ is s_CLOSED for all $l < j$, $g_s(x_j) = g_s^*(x_j)$ and similarly for $x_k$. Assuming S(x) = x_OPEN we have $f_s(x_j) = g_s(x_j) + h_s(x_j) \leq g_s^*(x_j) + H(x_j, x_k) + g_t(x_k) \leq g_s^*(x_j) + H^*(x_j, x_k) + g_t^*(x_k) = L$. The argument for S(x) = $\{PT_x(y) : y \in$ x_OPEN} uses $x_{k+1}$ instead of $x_k$ if $x_k \neq t$. The argument for $f_t$ is similar.

Now assume (iia). The reason we don't need (b1) to assure that not all $x_p \in \mu$ are x_CLOSED is that, due to GP's step (2.2a), if $\bar{x}$ were x_CLOSED it would also be $\bar{x}$_CLOSED causing $\mu$ to be discovered in step (2.3). The reason we don't need (b2) to assure that $j \leq k$ is that steps (2.2a), (2.2b), (2.2c), together, assure us that x_OPEN $\cap$ $\bar{x}$_CLOSED = $\emptyset$. The rest of (iia) is like (iib). The completes the lemma's proof.

To prove the admissibility theorem for GP we must eliminate the three cases mentioned above. Cases (1) and (2) are handled as in the proof of theorem 1 in [1]. To dispose of case (3) we must show that it is impossible for GP to halt with a non-optimal solution. Suppose that, on the contrary, GP halts with a path of cost $L' > L$. Then AMIN=L'. GP has not found a path of cost less than L' because otherwise AMIN would be less than L' and the corresponding path would be reported in step (3). But then, by the lemma, when GP finished its last outer loop there were nodes $m(x) \in$ x_OPEN satisfying $f_x(m(x)) \leq L < L'$ = AMIN, x=s,t. This is impossible because then the halting condition at step (2) could not be triggered. Thus case (3) is impossible. This completes the admissibility proof.

### 3.2 Admissible Pruning

The final GP search stage begins when some solution is found, at which point AMIN becomes finite. One may now prune x_OPEN $\cup$ x_CLOSED reducing target set sizes and the amount of list processing: It can be shown that, under assumptions (i) or (ii) of section 3.1, GP remains admissible if (a) new nodes with $f_x$-values $\geq$ AMIN are not kept, and (b) old nodes with $f_x$-values $\geq$ AMIN are removed from x_OPEN $\cup$ x_CLOSED.

## 4. A FIRST ORDER COMPARISON

In order to get a first order comparison of the total run time of several bidirectional search algorithms and unidirectional $A^*$ (UNI) the worst case behavior of these algorithms was analyzed in a particular search space. The results are summarized here. To a first approximation the run time may be written as an expression of the form $\alpha N + \beta H + \gamma L$, where $\alpha, \beta, \gamma$ are problem specific parameters, N is the number of nodes expanded, H is the number of H-calculations performed, and L is the total length of all lists searched. For example, if H-calculations are cheap while node expansion requires a lot of computer time, then $\alpha$ should be large and $\beta$ small. Our analysis focuses on the values of N, H, L for several algorithms.

The search space used was also studied by Champeaux and Sint [3] and Pohl [7;Chapter 7], all of whom calculated N for several algorithms: Let G be an undirected graph containing a countable collection of nodes; two nodes, s, t, have b edges ($b > 1$) and there is a path of length K between them. From all other nodes emanate b+1 edges. There are no cycles and all edge costs are one.

We have tabulated data about four algorithms: UNI, X, GP1 and GP2. X is a bidirectional search obtained by performing UNI on G x G. It is apparently due to Doran [4] and we use the precise description found in [2;section 2.1]. GP1 is a version of GP which uses ordered search, skips steps (2.2), (2.4) and sets x_TESTi = x_OPEN, i=1,2. We assume the front-to-front heuristic (1) of section 2.3 and alternating direction. When direction is changed GP1 never recalculates H-values. Instead it searches a matrix it maintains of relevant H-values. This method was used in a program by Champeaux and Sint [3]. We assume that either H(u,v) = H(v,u) or, if not, H returns both values.

We include GP2 to illustrate what happens when crucial changes are made in GP1. It is like GP1 except that a smaller target set is used and it handles differently the problem of updating f-values on OPEN when direction changes. When a new node is generated its H-values are calculated against the target set $S(x) = \{PT_x(y): y \in x\_OPEN\}$. Suppose we change direction to $\bar{x}$ and must now obtain the new f-values for nodes on $\bar{x}$_OPEN. Instead of recalculating H(u,v) for all $u \in \bar{x}\_OPEN$, $v \in S(x)$, we update each $h_{\bar{x}}(u)$ with respect to the new members of S(x) that were added since we were last going this direction. The effect is that $h_{\bar{x}}(u)$ is being calculated with respect to a target set larger than S(x), but this does not effect admissibility. Old nodes have f-values which are a little out-of-date but, if they are erroneously expanded, the children will have accurate information, hopefully preventing the faulty behavior from continuing. The purpose of this is to cut down on the list processing GP1 does to maintain its matrix of H-values.

Table 1 shows H, L values for the various algorithms in terms of N. We have kept only the highest order terms in N so the entries reflect assymptotic behavior. The X entries are closely related to the UNI entries because X behaves essentially like UNI with a branching factor of $b^2$ instead of b. The H(GP2) calculations were made using a worst case assumption that $|\{PT_x(y): y \in x\_OPEN\}| = |x\_CLOSED|$. The smaller GP2 target set unsurprisingly caused H(GP2) to be smaller than H(GP1) by essentially a factor of b. The casual update procedure for GP2 versus GP1 reduces the list processing from $O(N^3)$ to $O(N^2)$. While these appear good, one must remember that one or both may significantly reduce the heuristic power of GP2 causing N(GP2) to be greater than N(GP1). We have only begun empirical studies which would reveal if this is true.

| TABLE 1 | | GP1 | GP2 | X | UNI |
|---|---|---|---|---|---|
| | H | $(b^2-b)N^2/4$ | $(2b-1)N^2/4$ | $b^2N$ | $bN$ |
| | L | $b^2N^3/12$ | $(b^2+b-1)N^2/2$ | $(b^4+b^2-1)N^2/2$ | $(b^2+b-1)N^2/2$ |

| TABLE 2 | | Perfect Knowledge | $H^* \cdot / (1+\delta)$ | $H^* \pm \delta$ | No Knowledge |
|---|---|---|---|---|---|
| | GP | K | $2b^{(K-1)\delta-\beta}$ | $Kb^{[\delta]}$ | $2b^{(K/2)-1}$ |
| | X | K/2 | $b^{K\delta-2\beta}$ | $(K/2)b^{[\delta]}$ | $b^K$ |
| | UNI | K | $b^{k\delta-\beta}$ | $Kb^{[\delta]}$ | $b^K$ |

| TABLE 3 | | Perfect Knowledge | $H^* \cdot / (1+\delta)$ | $H^* \pm \delta$ | No Knowledge |
|---|---|---|---|---|---|
| | N | X<GP=UNI | GP<X<UNI | X<GP=UNI | GP<<X=UNI |
| | H | UNI$\leq$X<<GP | X<UNI<<GP if $\delta > \sqrt{2}$<br>UNI$\leq$X<<GP if $\delta \leq \sqrt{2}$ | X<UNI<<GP | GP<UNI<X |
| | L | GP=UNI<X | X<UNI $\Leftrightarrow \delta > \sqrt{2}$<br>GP<UNI $\Leftrightarrow \delta > 1/2$<br>GP < X | GP=UNI<X | GP<<UNI<X |

Table 2 shows the value of N for different algorithms and different heuristic situations. We have kept only the highest order terms in K (path length). Also expressions of the form $b^{aK}/(b^{a}-1)$ have been approximated by $b^{(a-1)K}$. GP represents either GP1 or GP2 since both have the same N values given our heuristic assumptions. Our heuristic assumptions are as follows: Columns 1, 4, respectively, assume perfect ($H=H^*$) and no ($H\equiv 0$) heuristic information. (In the latter case we assumed K even and that the solution is found as late as possible.) Columns 2, 3 assume worst case bounded error; column 2 is relative:

$$H(m,n) = \begin{cases} H^*(m,n)\cdot(1+\delta), & \text{if both } m, n \text{ are on the solution path} \\ H^*(m,n)/(1+\delta), & \text{otherwise;} \end{cases}$$

column 3 is absolute:

$$H(m,n) = \begin{cases} H^*(m,n) + \delta, & \text{if both } m, n \text{ are on the solution path} \\ H^*(m,n) - \delta, & \text{otherwise.} \end{cases}$$

We assume $\delta > 0$, $\beta = \delta(\delta+1)/(\delta+2)$, and $[\delta]$ is the integer part of $\delta$. We were surprised at how closely all the algorithms performed except when $H\equiv 0$; in this case GP excells. We were also surprised at the good performance of X.

Table 3 summarizes how the algorithms compare relative to N,H,L in various heuristic situations. The results are obtained by substituting Table 2 entries into Table 1. We set GP = GP2 since its Table 1 entries are best. If A,B are algorithms then A < B means that B performs worse than A by a constant factor; A<<B means that B performs worse than A by a factor that grows exponentially in K.

In Table 3 GP generally performs as well or better than UNI with respect to N and L. Unsurprisingly, the problem lies in time spent doing H-calculations. The table suggests that when heuristics are worse than bounded error we could expect GP to perform better than UNI with respect to N,L and comparably with respect to H. In only a very few entries does UNI beat both X and GP. This plus the better Table 1 performance of GP2 over GP1 suggests two directions for improving bidirectional search: Look for smaller accessible admissible target sets and combine the ideas of GP with those of X.

Appendix Proof that (2) is optimistic and that (3) is optimistic when $S(x)=\{x\} \cup x\_CLOSED$; H is assumed optimistic.

Consider the case of (2) first. Take $z \in G$. If z is not connected to $\bar{x}$ then $h^*_x(z)=\infty$ and there is nothing to prove. Otherwise let $\bar{x}=y_0,\ldots,y_n=z$ be an optimal path connecting z, $\bar{x}$. If $y_i \in \bar{x}\_CLOSED$ for all i, then $g_{\bar{x}}(z)=g^*_{\bar{x}}(z)$ so $h_x(z) \leq g_{\bar{x}}(z) = g^*_{\bar{x}}(z) = h^*_x(z)$, as is desired. Otherwise let j be least such that $y_j \in \bar{x}\_OPEN$. Then $g_{\bar{x}}(y_j) = g^*_{\bar{x}}(y_j)$ so $h_x(z) \leq H(z,y_j) + g_{\bar{x}}(y_j) \leq H^*(z,y_j) + g^*_{\bar{x}}(y_j) = h^*_x(z)$, since $y_j$ is on an optimal path between z and $\bar{x}$.

The proof of (3) when $S(x)=\{x\} \cup x\_CLOSED$ is similar except that the role of $y_j$ is now played by $y_{j-1}$, or $\bar{x}$ if j=0.

BIBLIOGRAPHY

[1] de Champeaux, D., Bidirectional heuristic search again, J. ACM, Vol. 30, 1983 (22-32).

[2] de Champeaux, D., and Sint, L., An improved bi-directional heuristic search algorithm, IJCAI, 1975 (309-314).

[3] de Champeaux, D., and Sint, L., An improved bi-directional heuristic search algorithm, J. ACM, Vol. 24, 1977, (177-191).

[4] Doran, J., Double tree searching and the graph traverser, Res. Memo EPU-R-22, Dept. of Machine Intelligence and Perception, Edinburgh University, Scotland, 1966.

[5] Hart, P., Nilsson, N., and Raphael, B., A formal basis for the heuristic determination of minimum cost paths, IEEE Transactions on Syst. Science and Cybernetics, SSC-4(2), 1968 (100-107).

[6] Nilsson, N., Principles of Artificial Intelligence, Tioga Publishing Co., Palo Alto, CA, 1980.

[7] Pohl, I., Bi-directional and heuristic search in path problems, SLAC Report No. 104, Stanford Linear Accelerator Center, Stanford, CA, 1969.

[8] Pohl, I., Bidirectional search, Machine Intelligence, Vol. 6, edited by B. Meltzer and D. Michie, 1971 (127-140).