Optimally Efficient Bidirectional Search

Eshed Shaham[1*], **Ariel Felner**[2], **Nathan R. Sturtevant**[3] and **Jeffrey S. Rosenschein**[1]
[1]School of Engineering and CS, Hebrew University of Jerusalem, Jerusalem, Israel
[2]ISE Department, Ben-Gurion University, Be'er Sheva, Israel
[3]CS Department, University of Alberta, Canada
eshed.shaham@mail.huji.ac.il, felner@bgu.ac.il, nathanst@ualberta.ca, jeff@cs.huji.ac.il

Abstract

A* is optimally efficient with regard to node expansions among unidirectional *admissible algorithms*—those that only assume that the heuristic used is admissible. This paper studies algorithms that are optimally efficient for bidirectional search algorithms. We present the Fractional MM algorithm and its sibling, the MT algorithm, which is simpler to analyze. We then develop variants of these algorithms that are optimally efficient, each under different assumptions on the information available to the algorithm.[1]

1 Introduction and Background

Given a graph G, the shortest-path problem is to find the least-cost path from state $start$ to state $goal$ in G. The traditional approach is to use the A* algorithm [Hart *et al.*, 1968]. A* is a unidirectional best-first search that priorities nodes according to $f(n) = g(n)+h(n)$ where $g(n)$ is the shortest known path from $start$ to the node n and $h(n)$ is an admissible heuristic (lower bound) on the cost of the path from n to the $goal$.

Bidirectional heuristic search algorithms (denoted henceforth by Bi-HS) interleave two separate searches, a search forward from s and a search backward from g. these algorithms usually halt once both frontiers include the same node. Bi-HS algorithms differ on which node they expand next and when exactly they stop such that optimality is guaranteed (see a survey in [Holte *et al.*, 2017]).

Recent research defined conditions on the node expansions required by Bi-HS algorithms to guarantee solutions optimality [Eckerle *et al.*, 2017]. Following work reformulated these conditions as a *must-expand graph* (G_{MX}), showing that the *Minimum Vertex Cover* (MVC) of G_{MX} corresponds to the minimal number of expansions required to prove optimality [Chen *et al.*, 2017]. In this paper we study the G_{MX} structure, characterize properties of the MVC and develop algorithms that expand exactly the MVC.

Let $d(x, y)$ denote the shortest distance between x and y and let $d(start, goal) = C^*$.

We use f_F, g_F and h_F to indicate f-, g-, and h-values in the forward direction and similarly, f_B, g_B and h_B to indicate f-, g-, and h-values in the backward direction. The *forward heuristic*, h_F, is *forward admissible* iff $h_F(u) \leq d(u, goal)$ for all u in G and is *forward consistent* iff $h_F(u) \leq d(u, u')+ h_F(u')$ for all u and u' in G. The *backward heuristic*, h_B, is *backward admissible* iff $h_B(v) \leq d(start, v)$ for all v in G and is *backward consistent* iff $h_B(v) \leq d(v', v) + h_B(v')$ for all v and v' in G. *Front-to-end* algorithms use these two heuristic functions coupled with the g-value of the corresponding node (v or u). Front-to-front bidirectional search algorithms use heuristics between pairs of states on opposite frontiers coupled with their respective g-values.

Let I_{AD} be the set of solvable problem instances in which the heuristics are *admissible*, and let I_{CON} be the subset of problems in I_{AD} where the heuristics are *consistent*. *Admissible algorithms* are algorithms that must be able to optimally solve *all* instances from I_{AD}. Admissible algorithms have no further knowledge about the problem or the heuristic (besides the fact that it is admissible), and they do not exploit such knowledge even if it is available. We deal with three different settings distinguished by the assumptions on the algorithms and the problem instances.

Case 1: Recent papers on bidirectional search (e.g., [Eckerle *et al.*, 2017]), followed the classic analysis of A* [Dechter and Pearl, 1985] and analyzed *admissible* algorithms running on problem instances with *consistent* heuristics. That is, the behavior of admissible algorithms was only studied when running on instances from I_{CON}. We denote these assumptions as the *base case* and label it by I_{AD}/I_{CON} along the following notation: *What can be assumed by the algorithm on the problem instances / The instances the algorithm is executed on*.

Case 2: In many domains (e.g., with unit-cost edges) the cost of the smallest edge (ϵ) is known. We label this case, where the algorithm knows ϵ and is allowed to exploit this knowledge as $I_{AD\epsilon}/I_{CON\epsilon}$ and show the adaptations that are needed to generalize existing theory and algorithms from I_{AD}/I_{CON} to $I_{AD\epsilon}/I_{CON\epsilon}$. Nevertheless, both of these cases have an inherent discrepancy — the assumptions on the knowledge available to the algorithm and the assumptions on the problem instances it will run on are different. This is the main motivation for the next case.

Case 3: The algorithm is allowed to exploit the fact that

*Contact Author

[1]This paper is a is a summary of two papers [Shaham et al. 2017; 2018] which are the basis for the M.Sc thesis of the first author.

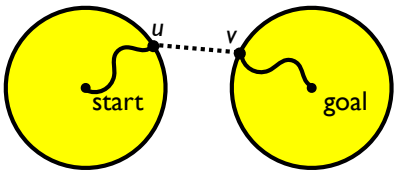

Figure 1: A path from $start$ to $goal$ that goes via u and v

both the forward and backward heuristics (h_F and h_B) are consistent. We label this case I_{CON}/I_{CON}. In this case algorithms can assume that they are running on instances from I_{CON} and we show that they can exploit a *front-to-front* heuristic that is induced by the consistency of the underlying heuristics.

1.1 Necessary Expansions in Bidirectional Search

Given an admissible heuristic, any admissible unidirectional algorithm must expand all states with $f(n) < C^*$ in order to prove a solution is optimal [Dechter and Pearl, 1985].

A new line of research was initiated by Eckerle *et al.* [2017] who generalized this theory to Bi-HS showing that necessary expansions are defined on pairs of states u and v in the forward and backward frontiers, respectively. Here we need to reason whether there can be an optimal path that goes from $start$ to u to v to the $goal$, as depicted in Figure 1. Three conditions are required to reason about potential paths between u and v:

1. $f_F(u) < C^*$
2. $f_B(v) < C^*$
3. $g_F(u) + g_B(v) < C^*$

If all conditions are met, the algorithm must explore to see if there is a shorter path between u and v. In bidirectional search, a pair of states (u, v) is called a *must-expand pair* if all three conditions are met. Unlike unidirectional search, in a *must-expand pair* only one of u or v must be expanded, not both. An algorithm that does not expand either u or v cannot be admissible because it may not find an optimal solution.

1.2 The Must-Expand Graph (G_{MX})

The unique property of the must-expand condition is that it contains an *or* between the nodes, also a feature of the vertex cover problem. Thus, the conditions for necessary expansions can be represented as the problem of finding a vertex cover on the *must-expand graph* denoted by G_{MX} [Chen *et al.*, 2017].

Definition 1. *The Must-Expand Graph G_{MX} on a problem instance is an undirected, unweighed bipartite graph. For each state $u \in G$, there are two vertices in G_{MX}, the left vertex u_F and the right vertex u_B. For each pair of states $u, v \in G$, there is an edge in G_{MX} between u_F and v_B if and only if (u, v) is a* must-expand pair.

It follows that the minimum number of node expansions required to solve a problem using Bi-HS is determined by the size of the minimum vertex cover of G_{MX} (denoted hereafter as MVC). Naturally, the exact set of vertices in G_{MX} depends on the heuristic used in the problem instance. Therefore, MVC is heuristic dependent as well.

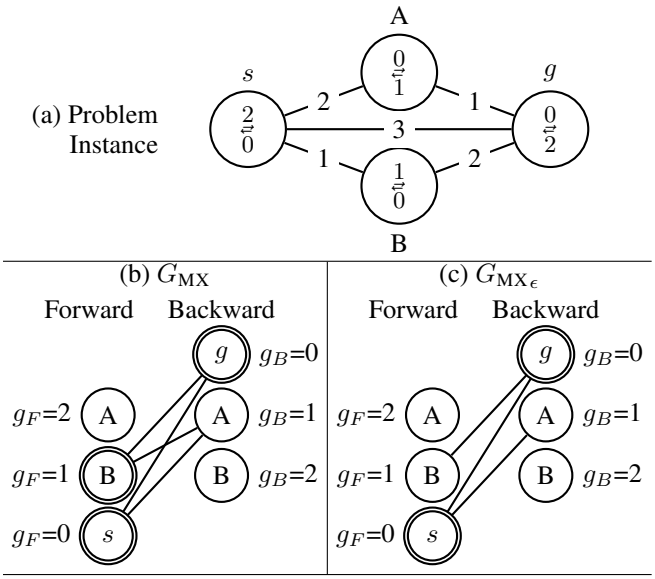

Figure 2: Case Study - Different versions of G_{MX}

Figure 2(b) shows the G_{MX} graph for the problem instance in Figure 2(a) where the numbers inside nodes are the h-values. The left vertices are sorted by increasing g_F-costs, while the right vertices are sorted by decreasing g_B-cost. In this ordering, each horizontal pair of states u and v has $g_F(u) + g_B(v) = C^*$. $C^* = 3$ in Figure 2.

2 MVC of G_{MX} in the I_{AD}/I_{CON} case (Case 1)

The MVC of G_{MX} has the following properties [Shaham *et al.*, 2017]. First, the MVC is *contiguous* in each direction. That is, if a node n with $g_F(n) = k$ is in the MVC, then all nodes with $g_F(n) \leq k$ in G_{MX} must also be in the MVC, and similarly for g_B. Second, there exist thresholds t_F^*, t_B^* such that (1) $t_F^* + t_B^* = C^*$ and (2) a forward node u from G_{MX} is included in the MVC iff $g_F(u) < t_F^*$, and a backward node v from G_{MX} is included in the MVC iff $g_B(v) < t_B^* = C^* - t_F^*$. In Section 3 we will generalize this to Case 2 ($I_{AD\epsilon}/I_{CON\epsilon}$).

To demonstrate this, consider all pairs of values of t_F and t_B where $t_F + t_B = C^*$. There is a family of contiguous vertex covers (VCs) for all such pairs (t_F, t_B), where all nodes with $g_F < t_F$ in G_{MX} are included in the forward direction of this VC, and all nodes with $g_B < t_B$ are included in the backward direction of this VC. One such (t_F, t_B) partition is the MVC and is denoted by (t_F^*, t_B^*). The number of nodes of the (t_F, t_B) VC partitions is determined by summing up the nodes with $g_F < t_F$ and with $g_B < t_B$. In Figure 2, nodes that correspond to each (t_F, t_B) pair are aligned horizontally. Nodes that are included in MVC have a double outline — these are nodes with $g_F < t_F^* = 2$ and with $g_B < t_B^* = 1$.

2.1 Fractional MM

MM is a Bi-HS algorithm that is guaranteed to *meet in the middle* [Holte *et al.*, 2017]. That is, MM will never expand a state

whose g-value exceeds $C^*/2$. *Fractional MM* (fMM) generalizes MM and can meet at any fraction of the optimal solution cost [Shaham *et al.*, 2017]. fMM(p) uses the following priority functions on nodes in the open lists, where $0 < p < 1$:

$$pr_F(u) = \max(g_F(u) + h_F(u), g_F(u)/p)$$
$$pr_B(v) = \max(g_B(v) + h_B(v), g_B(v)/(1-p))$$

fMM(p) expands a state with minimum priority in either direction. fMM(p)'s forward and backward searches meet at $(p \cdot C^*, (1-p) \cdot C^*)$ by similar reasoning for why MM meets in the middle. That is, fMM(p) will never forward expand a state whose g_F-value exceeds $p \cdot C^*$ and never backward expand a state whose g_B-value exceeds $(1-p) \cdot C^*$. This attribute is called *restrained with respect to* p. As a result, a restrained algorithm will always return the optimal solution because it can only meet at the meeting point along the optimal path. MM is a special case of fMM(p) for $p = 1/2$. Forward A* and reverse A* (searching from *goal* to *start*) are also special cases corresponding to $p = 1$ and $p = 0$ respectively.

For every problem instance, there exists a fraction $0 \leq p^* \leq 1$ such that fMM(p^*) is *optimally efficient* and will expand the minimal number of necessary nodes — those in the MVC [Shaham *et al.*, 2017]. In practice, however, the exact value for p^* is not known a priori, as it depends on C^* and G_{MX}. Both are not known in advance before the execution of the algorithm. But, given C^* and G_{MX} it is possible to find p^* in time linear in the number of distinct g-values. Finding p^* and then running fMM(p^*) can serve for research purposes as an oracle—any algorithm can now be compared to the theoretical optimum.

The definition of fMM through fractions resulted from the attempt to generalize existing algorithms such as MM and A*. Since the MVC theory uses thresholds it is more natural to use an algorithm that uses such thresholds.

2.2 Meet at the Threshold

Given a threshold t, the *meet at the threshold* algorithm (MT) [Shaham *et al.*, 2018] is *restrained* with respect to t, or equivalently, MT meets at $(t, C^* - t)$. That is, MT will never forward expand a state whose g_F-value exceeds t and never backward expand a state whose g_B-value exceeds $C^* - t$. MT(t) uses the following priority functions:

$$pr_F(u) = \begin{cases} g_F(u) + h_F(u) & \text{if } g_F(u) < t \\ \infty & \text{if } g_F(u) \geq t \end{cases}$$

$$pr_B(v) = \max(g_B(v) + h_B(v), g_B(v) + t)$$

The same line of proof for MM and fMM fits here as well. The fact that MT is restrained ensures an optimal solution.

For $t = 0$, MT(t) is identical to backward A*. Similarly, for $t \geq C^*$, MT(t) is identical to forward A*. It is easy to see that MT(t) will expand the same set of nodes in G_{MX} (not necessarily in the same order) as fMM(p) when $t = p \cdot C^*$. For example, MM is identical to MT(t) for $t = C^*/2$. Therefore, for any value of t, MT will return the optimal solution. Moreover, when choosing the optimal value for t ($t^* = p^* \cdot C^*$), MT(t) is also *optimally efficient*, i.e., it expands exactly the nodes in the MVC. In the next sections we study G_{MX} and its MVC, and generalize MT (and fMM, where possible) to the cases where the algorithm can assume more knowledge in the graph and on h.

2.3 Other G_{MX}-based Algorithms

The structure of G_{MX} is not given as input and is not known in advance. NBS [Chen *et al.*, 2017] is a Bi-HS algorithm that aims to find some VC of G_{MX} by picking a and expanding both its vertices. Similarly, DVCBS [Shperberg *et al.*, 2019] is a Bi-HS algorithm that grows G_{MX} on the fly and always expands the MVC of the growing G_{MX}. NBS has a guarantee that the size of its VC is at most twice the size of the MVC. DVCBS does not have that guarantee but had better average case performance.

3 Case 2: Knowing ϵ ($I_{AD\epsilon}/I_{CON\epsilon}$)

In many cases, the problem instance is coupled with a lower bound (ϵ) on the edge costs. For example, in unit edge-cost domains $\epsilon = 1$. In other domains, one might iterate over all actions costs and take their minimum. This bound provides an admissible front-to-front heuristic of ϵ between any two distinct nodes. We now study the case where algorithms are allowed to assume the existence of ϵ but as done in previous work can still only assume an admissible heuristic. Similarly, we assume that they are evaluated in instances where the heuristic is consistent. In our notation, this case is labeled by $I_{AD\epsilon}/I_{CON\epsilon}$. For this case we adapt the conditions for *must-expand pairs* in domains with front-to-front heuristics as defined by Eckerle *et al.* [2017] with ϵ as the front-to-front heuristic. In such cases two nodes u, v are a *must-expand pair* if the following conditions are met:

1. $f_F(u) < C^*$
2. $f_B(v) < C^*$
3. $g_F(u) + g_B(v) + \epsilon < C^*$

Condition 3 is more informative than the front-to-end version of the I_{AD}/I_{CON} case (Case 1) due to the addition of ϵ. In Figure 1, ϵ is a lower bound on the cost of the path from u to v. Therefore, some edges that exist in the old G_{MX} no longer exist in the new G_{MX}. Figure 2(c) shows such a G_{MX} for the instance in Figure 2(a). In this instance, $\epsilon = 1$ and thus (B, A) is no longer a *must-expand pair*, since $g_F(B) + g_B(A) + \epsilon = 1 + 1 + 1 = C^*$.

Similar to the I_{AD}/I_{CON} case we show that there exist thresholds t_F^*, t_B^* such that a forward node u from G_{MX} is included in MVC iff $g_F(u) < t_F^*$, and a backwards node v from G_{MX} is included in MVC iff $g_B(v) < t_B^*$. The main difference being that this time $t_F^* + t_B^* + \epsilon = C^*$.

We now define the following four terms:

Definition 2.

$$g_{F^{\text{I}}} = \max_{u \in (G_{\text{MX}_F} \cap \text{MVC})} \{g_F(u)\}$$
$$g_{F^{\text{O}}} = \min_{u \in (G_{\text{MX}_F} \setminus \text{MVC})} \{g_F(u)\}$$
$$g_{B^{\text{I}}} = \max_{v \in (G_{\text{MX}_B} \cap \text{MVC})} \{g_B(v)\}$$
$$g_{B^{\text{O}}} = \min_{v \in (G_{\text{MX}_B} \setminus \text{MVC})} \{g_B(v)\}$$

Based on these terms we define the following thresholds:

$$t_F^* = \frac{\max\{g_{F^{\text{I}}}, C^* - \epsilon - g_{B^{\text{O}}}\} + \min\{g_{F^{\text{O}}}, C^* - \epsilon - g_{B^{\text{I}}}\}}{2}$$

$$t_B^* = \frac{\max\{g_{B^{\text{I}}}, C^* - \epsilon - g_{F^{\text{O}}}\} + \min\{g_{B^{\text{O}}}, C^* - \epsilon - g_{F^{\text{I}}}\}}{2}$$

Note that $t_F^* + t_B^* + \epsilon = C^*$. Furthermore, these thresholds indeed satisfy the following theorem.

Theorem 1. *For a forward node u, $u \in$ MVC iff $g_F(u) < t_F^*$. For a backward node v, $v \in$ MVC iff $g_B(v) < t_B^*$.*

For Figure 2(c) we have $t_F^* = 1$ and $t_B^* = 1$ (as opposed to the I_{AD}/I_{CON} case which had $t_F^* = 2$).

Theorem 1 implies a simple algorithm for finding MVC given that G_{MX} and C^* are both known. We iterate over all possible thresholds that satisfy $t_F + t_B + \epsilon = C^*$ and sum up the costs of the forward nodes $u \in G_{\text{MX}}$ for which $g_F(u) < t_F$ as well as the costs of the backward nodes $v \in G_{\text{MX}}$ for which $g_B(v) < t_B$. This algorithm runs with complexity linear in the number of distinct g-values of states in G_{MX}. It is similar to the *Calc-VC()* algorithm provided by Shaham *et al.* [2017].

3.1 Meet at the Threshold ϵ (MTϵ)

The priority function of MT can easily be adapted to create a new algorithm MTϵ:

$$pr_F(u) = \begin{cases} g_F(u) + h_F(u) & \text{if } g_F(u) < t_F \\ \infty & \text{if } g_F(u) \geq t_F \end{cases}$$
$$pr_B(u) = \max(g_B(u) + h_B(u), g_B(u) + t_F + \epsilon)$$

Similar to MT(t) in the I_{AD}/I_{CON} case, for any given problem instance I in $I_{CON\epsilon}$ there exists t_F^* such that MT$\epsilon(t)$ is optimally efficient—it expands the minimal number of nodes in G_{MX} of I. The proof for this is nearly identical to the respective proof on fMM.

4 Case 3: Consistency I_{CON}/I_{CON}

As explained above, the analysis of Bi-HS by [Eckerle *et al.*, 2017] and subsequent papers [Chen *et al.*, 2017; Shaham *et al.*, 2017] assumed admissible algorithms. However, they all analyzed the case where such algorithms are executed on instances from I_{CON}. We next investigate the case where the algorithms know that the heuristics are consistent (i.e., they know that the instances are from I_{CON}) and they are allowed to exploit that knowledge.

Given consistent heuristics, $|h_F(u) - h_F(v)|$ is a lower bound on the distance between u and v, so $|h_F(u) - h_F(v)|$ can be used as a front-to-front heuristic between u and v. Similarly, $|h_B(u) - h_B(v)|$ can be used as a heuristic as well. This was called the *Add* method by Kaindl and Kainz [1997].

Definition 3. (Front-to-front heuristic) *For each pair of nodes (a, b) define the following admissible heuristic:*

$$h_C(a, b) = \max\{|h_F(a) - h_F(b)|, |h_B(a) - h_B(b)|\}$$

This heuristic results from the combination of assuming both forward and backward consistency (hence h_C).

4.1 Necessary Node Expansions

Using the conditions for *must-expand pairs* [Eckerle *et al.*, 2017] in problems with h_C as a front-to-front heuristic results in the following claim: two nodes u, v are a *must-expand pair* if the following conditions are met:

1. $f_F(u) < C^*$
2. $f_B(v) < C^*$
3. $g_F(u) + g_B(v) + h_C(u, v) < C^*$

Any pair of nodes u, v that satisfy these conditions is a *must-expand pair* and induces an edge in G_{MX} between u and v. Condition 3 is more informative than the front-to-end version due to the addition of h_C. Therefore, some edges that exist in the old G_{MX} no longer exist in the new G_{MX}.

4.2 Equivalence Classes

In the I_{AD}/I_{CON} case $h_C(u, v)$ did not exist. To imitate this in the I_{CON}/I_{CON} case, we partition G_{MX} into equivalence classes, such that any given pair of nodes in each class (u, v) has $h_C(u, v) = 0$.

Definition 4. (heuristic equivalence) *Two nodes u, v are heuristically equivalent (or h-equivalent) iff $h_C(u, v) = 0$.*

Directly from the definition, two nodes u, v are h-equivalent (belong to the same h-equivalence class) iff $h_F(u) = h_F(v)$ and $h_B(u) = h_B(v)$.

Lemma 2. (Restrained with respect to t_{FQ}^*) *For every h-equivalence class Q there exist thresholds $t_{FQ}^* + t_{BQ}^* = C^*$ such that MVC $\cap\, Q$ contains exactly the forward nodes in Q for which $g_F < t_{FQ}^*$. and exactly all backward nodes for which $g_B < t_{BQ}^*$.*

Another way of looking at this would be to describe t_{FQ}^* as a function of the h-values of Q:

Corollary 3. *There exists a function $b : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$ such that a forward node u is in MVC iff $g_F(u) \leq b(h_F(u), h_B(u))$ and a backward node v is in MVC iff $g_B(u) \leq C^* - b(h_F(u), h_B(u))$.*

Below we use $b(u)$ as shorthand for $b(h_F(u), h_B(u))$.
We can now adapt MT to work in the I_{CON}/I_{CON} case. We define the algorithm MT$_{CON}(b)$ with the following priorities:

$$pr_F(u) = \begin{cases} g_F(u) + h_F(u) & \text{if } g_F(u) < b(u) \\ \infty & \text{if } g_F(u) \geq b(u) \end{cases}$$
$$pr_B(u) = \max(g_B(u) + h_B(u), g_B(u) + b(u))$$

4.3 Attributes of b

As discussed above, given any threshold $t_F \in \mathbb{R}^+$, MT and MTϵ (and similarly, fMM and fMMϵ) will always return an optimal path, although they might do more work than the minimal number of necessary expansions (as achieved by t_F^*). MT$_{CON}$, however, is not guaranteed to return an optimal path for any $b : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$.

Fortunately this issue can be solved by constraining b to be a member of the following set of functions B:

$$B = \{b : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+ \mid \forall {h_F}', {h_B}', {h_F}'', {h_B}'' \in \mathbb{R}^+ \\ \left|b({h_F}', {h_B}') - b({h_F}'', {h_B}'')\right| \\ \leq \max\{\left|{h_F}' - {h_F}''\right|, \left|{h_B}' - {h_B}''\right|\}$$

Case	I_{AD}/I_{CON}	$I_{AD\epsilon}/I_{CON\epsilon}$	I_{CON}/I_{CON}
MVC	Restrained	ϵ-Restrained	$b \in B$
Optimal Alg.	fMM / MT	MTϵ / fMMϵ	$\mathtt{MT}_{CON}$
# Clusters	$\lvert\mathbb{G}\rvert$	$\lvert\mathbb{G}\rvert$	$\lvert\mathbb{G}\rvert \times \lvert\mathbb{H}\rvert^2$
Complexity	$O(\lvert\mathbb{G}\rvert)$	$O(\lvert\mathbb{G}\rvert)$	$O(\lvert\mathbb{G}\rvert^2 \times \lvert\mathbb{H}\rvert^4)$

Table 1: A summarizing table. $|\mathbb{G}|$ and $|\mathbb{H}|$ are the numbers of distinct g- and h-values respectively.

This is a stronger version of the global *1-Lipschitz* attribute on the continuity of functions. We prove two theorems:
(1:) Given a function $b \in B$, $\mathtt{MT}_{CON}(b)$ will return an optimal path on any problem instance in I_{CON}.
(2:) Given a problem instance in I_{CON} there exists $b^* \in B$ such that $\mathtt{MT}_{CON}(b)$ will expand the minimal number of necessary expansions on that instance.

From these theorems we can deduce that $\mathtt{MT}_{CON}$ is optimally efficient in the I_{CON}/I_{CON} case, provided that only functions $b \in B$ are allowed as the argument. A detailed analysis and related proofs are provided in [Shaham *et al.*, 2018].

5 Summary and Conclusions

We have shown the nature of MVC for G_{MX} for cases with different assumptions on the knowledge of the algorithm about the heuristics used and on the nature of the underlying graph. This enriches the theory on G_{MX} to more cases. We have also developed MT, which is equivalent to fMM but is simpler for our purposes.

Table 1 summarizes our findings. Each column represents a different case. The first row presents the structure of MVC. The second row presents the optimally efficient algorithm. The third row gives the number of clusters of indistinguishable nodes in G_{MX}, i.e., clusters that are either fully contained within an MVC or not at all. The complexity row gives the complexity of the best-known algorithm to calculate MVC given that G_{MX} and C^* are known. While an algorithm for the I_{AD}/I_{CON} case and the $I_{AD\epsilon}/I_{CON\epsilon}$ case were provided, for the I_{CON}/I_{CON} case one should resort to the known algorithm for finding a MVC in a bipartite graph [Papadimitriou and Steiglitz, 1982].

Future work can further expand this theory to more cases such as the case where we add ϵ to the I_{CON}/I_{CON} case. In addition, an important line will be to develop such a theory to the case where we have a full front-to-front heuristic.

6 Acknowledgments

The work was supported by the Israel Science Foundation (ISF) grant #844/17, by BSF grant #2017692, by NSF grant #1815660 and by the HUJI Cyber Security Research Center in conjunction with the Israel National Cyber Directorate (INCD) in the Prime Minister's Office.

References

[Chen *et al.*, 2017] Jingwei Chen, Robert C. Holte, Sandra Zilles, and Nathan R. Sturtevant. Front-to-end bidirectional heuristic search with near-optimal node expansions. In *Proceedings of IJCAI*, 2017.

[Dechter and Pearl, 1985] Rina Dechter and Judea Pearl. Generalized best-first search strategies and the optimality of A*. *J. ACM*, 32(3):505–536, 1985.

[Eckerle *et al.*, 2017] Jurgen Eckerle, Jingwei Chen, Nathan R. Sturtevant, Sandra Zilles, and Robert C. Holte. Sufficient conditions for node expansion in bidirectional heuristic search. In *ICAPS*, 2017.

[Hart *et al.*, 1968] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Systems Science and Cybernetics*, 4(2):100–107, 1968.

[Holte *et al.*, 2017] Robert C. Holte, Ariel Felner, Guni Sharon, Nathan R. Sturtevant, and Jingwei Chen. Bidirectional search that is guaranteed to meet in the middle. *Artificial Intelligence Journal (AIJ), In press*, 2017.

[Kaindl and Kainz, 1997] Hermann Kaindl and Gerhard Kainz. Bidirectional heuristic search reconsidered. *J. Artif. Intell. Res.*, 7:283–317, 1997.

[Papadimitriou and Steiglitz, 1982] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1982.

[Shaham *et al.*, 2017] Eshed Shaham, Ariel Felner, Jingwei Chen, and Nathan R. Sturtevant. The minimal set of states that must be expanded in a front-to-end bidirectional search. In *SoCS*, pages 82–90, 2017.

[Shaham *et al.*, 2018] Eshed Shaham, Ariel Felner, Nathan R. Sturtevant, and Jeffrey S. Rosenschein. Minimizing node expansions in bidirectional search with consistent heuristics. In *SOCS*, pages 81–98, 2018.

[Shperberg *et al.*, 2019] Shahaf Shperberg, Avi Hayoun, Ariel Felner, Solomon Eyal Shimony, and Nathan R. Sturtevant. Enriching non-parametric bidirectional search algorithms. In *AAAI*, 2019.