Performance Of IDA* on Trees and Graphs*

Ambuj Mahanti[†] Systems Res. Ctr. U. of Maryland College Park MD 20742

Subrata Ghosh[‡] Comp. Sci. Dept. U. of Maryland College Park MD 20742

Dana S. Nau[§] Comp. Sci. Dept. U. of Maryland College Park MD 20742

Asim K. Pal IIM, Calcutta Calcutta 700 027 India

Laveen Kanal[¶] Comp. Sci. Dept. U. of Maryland College Park MD 20742

Abstract

We present the following results about IDA* and related algorithms:

- We show that IDA* is not asymptotically optimal in all of the cases where it was thought to be so. In particular, there are trees satisfying all of the conditions previously thought to guarantee asymptotic optimality for IDA*, such that IDA* will expand more than $O(N)$ nodes, where N is the number of nodes eligible for expansion by A*.
- We present a new set of necessary and sufficient conditions to guarantee that IDA* expands $O(N)$ nodes on trees.
- On trees not satisfying the above conditions, there is *no* best-first admissible tree search algorithm that runs in $S = N/\psi(N)$ (where $\psi(N) \neq O(1)$) memory and always expands $O(N)$ nodes.
- There are acyclic graphs on which IDA* expands $\Omega(2^{2N})$ nodes.

Introduction

Heuristic search is applicable to a wide range of combinatorial optimization problems. The objective of many heuristic search algorithms is to find a minimum cost solution path in a directed graph G. A solution path is a path from the start node s to a goal node. To find such a path, many algorithms use a node evaluation function

$$f(n) = g(n) + h(n),$$

where $g(n)$ is the cost of a least-costly path currently known from s to n, and $h(n) \geq 0$, the heuristic value of node n, is an estimate of $h^*(n)$. h is called the heuristic function and $h^*(n)$ is the cost of a minimum cost path from n to a goal node. A heuristic function h is called admissible if $\forall n \in G$, $h(n) \leq h^*(n)$. The function h is said to be monotone if $\forall p \in G$, $h(p) \leq c(p,q) + h(q)$, where q is a child of p.

A* (Hart & Nilsson & Raphael, 1968; Nilsson, 1980) is a well-known heuristic search algorithm. A* has been shown to be very efficient in terms of number of node expansions (which is also a measure of its time complexity) in most cases (Dechter & Pearl, 1985). However, one major problem with A* is that it requires exponential amount of memory to run. Due to this, A* runs out of memory even on problem instances of moderate size.

To overcome the storage problem, a variant of A* called IDA* (Iterative Deepening A*) was introduced by Korf (Korf, 1985; Korf, 1988). IDA*'s memory requirement is only linear in the depth of the search. This enables IDA* to solve much larger problems than that A* can solve in practice.

One of IDA*'s most important properties is that under certain conditions it is "asymptotically optimal in time and space over the class of best-first searches that find optimal solutions on a tree" (Korf, 1988, p. 236).

In this paper, we present the following results:

1. We show that IDA* is not asymptotically optimal in all of the cases where it was thought to be so. In particular, there are trees satisfying all of asymptotic optimality conditions given in (Korf, 1988), such that IDA* will expand more than $O(N)$ nodes, where N is the number of nodes eligible for expansion by A*.[1] In addition, we present necessary and sufficient conditions for the desired $O(N)$ worst-case time complexity of IDA* for tree searches.

*Supported in part by NSF Grant IRI 8802419, NSF Grant NSFD CDR-88003012 to the University of Maryland Systems Research Center, NSF grant IRI-8907890, and CMDS project (Work order no. 019/7-148/CMDS-1039/90-91.)

[†]Also in the Computer Science Department. Email: am@cs.umd.edu

[‡]Email: subrata@cs.umd.edu

[§]Also in the Systems Research Center and the Institute for Advanced Computer Studies. Email: nau@cs.umd.edu

[¶]Email: kanal@cs.umd.edu

[1]Previous papers have described treees on which IDA* expands more than $O(N)$ nodes (Mahanti & Pal, 1990; Patrick & Almulla & Newborn, 1992), but the trees described in these papers did not satisfy requirements (Korf, 1988) of finite precision and non-exponential node costs.

2. In the absence of any assumptions (except that the heuristic is admissible and the maximum node-branching factor is constant) there does not exist any best-first admissible tree search algorithm which when running with $S = N/\psi(N)$ ($\psi(N) \neq O(1)$) memory can have the $O(N)$ worst-case time complexity like A*.
3. When the heuristic is monotone, A* handles a graph like a tree and it never expands a node more than once. But for graphs, IDA* can not prevent the reexpansion of a node through a costlier path. Thus, for graph search problems the performance of IDA* is bound to become worse than A*. We show that if we let the node-branching factor to grow with the problem size, A* under monotone heuristic has $O(N)$ worst-case time complexity for general graphs, but IDA* under monotone heuristics has $\Omega(2^{2N})$ worst-case complexity for acyclic graphs. And, the total number of node expansions by IDA* can only increase in presence of cycles. There are many graph and tree search problems where the node-branching factor grows with the problem size. Traveling salesman, flow-shop scheduling, etc. are such examples.

Due to space limitations, in this paper we omit the proofs of our theorems. For proofs, readers are referred to (Mahanti *et al.*, 1992).

IDA* on Trees

In this section we first define a set of basic symbols that will be used through out the paper, and then formally characterize the working of IDA*. Here we assume that the state space G is a tree, the maximum node-branching factor in G is bounded by a constant $b > 0$, and every arc of G has a cost $\geq \delta$, where δ is a small constant.

For each $z > 0$, we define $\mathcal{W}^G(z)$ as follows:

(i) $P = (s)$ is in $\mathcal{W}^G(z)$ if s is not a goal node and $h(s) \leq z$.

(ii) For each path $P = (s, n_1, \ldots, n_k)$ in G, P is in $\mathcal{W}^G(z)$ if the following conditions are satisfied:

(a) n_k is not a goal node,

(b) The subpath $P' = (s, n_1, \ldots, n_{k-1})$ is in $\mathcal{W}^G(z)$, and

(c) $\text{cost}(P) + h(n_k) \leq z$.

We also define

$$\begin{aligned} \mathcal{V}^G(z) &= \{m | m \text{ is a node in a path in } \mathcal{W}^G(z)\}; \\ \mathcal{N}^G(z) &= |\mathcal{V}^G(z)|. \end{aligned}$$

Since by assumption the maximum node-branching factor b is a constant, and each arc (m, n) in G has a cost at least δ, it directly follows that for each $z > 0$, each of the entities defined above is finite.

We define f_i, $i = 1, 2, \ldots$, inductively as follows:

$$\begin{aligned} f_1 &= h(s); \\ f_i &= \min_n\{f(n) | n \text{ is a child of tip}(P) \text{ and } P \text{ is a maximal path in } \mathcal{W}^G(f_{i-1})\}, \end{aligned}$$

where by a maximal path P in $\mathcal{W}^G(f_{i-1})$, we mean a path which is not a proper subpath of any other path in $\mathcal{W}^G(f_{i-1})$. Also, by tip(P) of a path P, we mean the last node on P.

We let I^G be the total number of iterations performed by IDA* on G, and

$$z^G(1) < z^G(2) < \ldots < z^G(I^G)$$

be the (distinct) threshold values used by IDA*. For $j = 1, 2, \ldots, I$, IDA*'s j'th *iteration* is the set of all node expansion instants for which the threshold is $z(j)$. By expansion of a node n in IDA*, we mean the generation of at least one child of n. For each j, we define

$$\begin{aligned} X^G(j) &= \text{the set of nodes expanded by IDA* during iteration } j; \\ x^G(j) &= \text{the number of nodes expanded by IDA* during iteration } j; \\ X^G_{\text{new}}(j) &= \text{the set of new nodes expanded by IDA* during iteration } j; \\ x^G_{\text{new}}(j) &= \text{the number of new nodes expanded by IDA* during iteration } j; \\ x^G_{\text{tot}} &= \text{the total number of node expansions done by IDA* on } G. \end{aligned}$$

In the terms defined above, we will usually omit the superscript G if the identity of G is clear. Alternatively, if we are discussing two state spaces G and G', we will use $X(j)$ for $X^G(j)$, $X'(j)$ for $X^{G'}(j)$, and so forth.

From the above definitions, it follows immediately that

$$X_{\text{new}}(j) = \begin{cases} X(1), & j = 1; \\ X(j) - X(j-1), & j = 2, 3, \ldots; \end{cases} \quad (1)$$

$$x_{\text{tot}} = \sum_{j=1}^{I} x(j) = \sum_{j=1}^{I} |X(j)|. \quad (2)$$

Theorem 1

$$z(j) = f_j, \quad j = 1, \ldots, I; \quad (3)$$

$$X(j) = \mathcal{V}(f_j), \quad j = 1, \ldots, I-1; \quad (4)$$

$$x(j) = \mathcal{N}(f_j), \quad j = 1, \ldots, I-1; \quad (5)$$

$$X_{\text{new}}(j) = \begin{cases} \mathcal{V}(f_1), & j = 1; \\ \mathcal{V}(f_j) - \mathcal{V}(f_{j-1}), & j = 2, \ldots, I-1; \end{cases} \quad (6)$$

$$x_{\text{new}}(j) = \begin{cases} \mathcal{N}(f_1), & j = 1; \\ \mathcal{N}(f_j) - \mathcal{N}(f_{j-1}), & j = 2, \ldots, I-1; \end{cases} \quad (7)$$

Furthermore,

$$X(I) \subseteq \mathcal{V}(f_I); \quad (8)$$

$$X_{\text{new}}(I) \subseteq \mathcal{V}(f_I) - \mathcal{V}(f_{I-1}); \quad (9)$$

$$x_{\text{new}}(I) \leq \mathcal{N}(f_I) - \mathcal{N}(f_{I-1}); \quad (10)$$

with equality in the worst case.

Corollary 1 $f_I = h^*(s)$.

In view of the above corollary, we make the following definitions:

$$\begin{aligned} W^G &= \mathcal{W}(f_I); \\ V^G &= \mathcal{V}(f_I); \\ N^G &= \mathcal{N}(f_I); \\ L^G &= 1 + \max_{P \in W^G} \text{the number of nodes in } P. \end{aligned}$$

In the above, we will omit the superscript G when the identity of G is clear. For example, given a network G, by N we shall mean the total number of nodes in G which are eligible for expansion by A*.

The *heuristic branching factor* is defined as the average, over $j = 2, \ldots, I$, of the quantity $x_{\text{new}}(j)/x_{\text{new}}(j-1)$. Intuitively, this is the average ratio of the number of nodes of each f-value (assuming that the heuristic is monotone) to the the number of nodes at the next smaller f-value (Korf, 1988).

Under the premise that G is a tree with maximum node-branching factor b, and with admissible heuristics, Korf (Korf, 1988) has shown that the worst-case asymptotic time complexity of IDA* is $O(N)$ if the following conditions (labelled as *mandatory* and *secondary*) are true:

Mandatory Condition:

Heuristic Branching Factor > 1.

Secondary Conditions:

1. *The search space must be exponential in the depth of the solution;*
2. *Representation of costs must be with finite precision;*
3. *Cost values must not grow exponentially with depth.*

The first condition was used explicitly in the optimality proof of IDA* (thus we call it a mandatory condition), and the other conditions appeared as passing remarks (thus we call them secondary conditions). In the next section we show that these conditions are neither sufficient nor necessary to ensure the $O(N)$ complexity of IDA*. We illustrate through examples that even when all of the above conditions are satisfied, IDA* fails to achieve $O(N)$ time complexity in the worst-case.

IDA* on Example Trees

In this section we illustrate through examples that the analysis of IDA* given in (Korf, 1988) does not hold in general. We present constructions of example trees which satisfy the conditions stated in the previous section but yet IDA* fails to achieve $O(N)$ worst-case time complexity while run on these trees. We also show that these conditions are not necessary either, i.e. IDA* can have $O(N)$ complexity without satisfying these conditions.

Example 1. In the search tree G given in Figure 1, each non-leaf node has a node-branching factor $b = 2$, and each arc has unit cost. G consists of two subtrees (called G_1 and G_2) where each one is a full binary tree of height k. G_2 is rooted at the right most node of G_1. Every leaf node, except the one labelled as goal, is a non-terminal leaf node. For each node n in G, we assume $h(n) = 0$. Then h is monotone. The heuristic branching factor is

$$\frac{2k + \frac{1}{2^{k-1}} + 2(k-1)}{(2k)} = 2 + \frac{1}{k2^k} - \frac{1}{k} \approx 2.$$

Note that the goal node is at a depth of $2k = O(logN)$, where N is the total number of non-goal nodes in G. Therefore the search space is exponential. The maximum cost value is $2k$ which grows only linearly with depth. The precision constraint is vacuously satisfied because the cost values are not fractions. Thus, all conditions (mandatory and secondary) are satisfied. Now we calculate the total number of node expansions by IDA* on the tree G.

Clearly G_1 and G_2 each contain $N' = \lceil N/2 \rceil$ nodes. The cost of the solution path is $2k = 2[\log_2(N'+1) - 1]$. Let

$$N_0 = b^k + 2b^{k-1} + 3b^{k-2} + \ldots + kb.$$

Then the total number of node expansions by IDA* in the worst-case is

$$\begin{aligned} x_{tot} &= N_0 + kN' + N_0 \\ &\geq kN' + N' = k(N'+1) \\ &= \Omega(N \log N). \end{aligned}$$

In the example above, we have shown that the conditions stated in (Korf, 1988) for the asymptotic optimality of IDA* are not sufficient. In the following example we show that these conditions are not necessary either.

Example 2. Consider the search tree G in Figure 2. G consists of two subtrees G_1 and G_2. G_1 is a full binary tree with N' nodes and G_2 contains a constant c number of nodes in the form of a degenerate tree. Every leaf node in G is a non-terminal node except the rightmost one (p_c), which is a goal node. Each arc has cost 1, $h(s) = k$ and heuristic value at all other nodes is zero. G contains total $N = N' + c - 1$ non-goal nodes and one goal node. All the nodes of G_1 will be expanded by IDA* in the first iteration. Thereafter, in each iteration only one new node will be expanded. The heuristic branching factor is

$$\frac{(c-1) * 1 + \frac{1}{N'}}{c} < 1.$$

Since the total number of iterations (c+1) is constant, IDA* will expand only $O(N)$ nodes on trees of type G. Note that the mandatory condition stated previously is not satisfied in this case.

In the following section we derive a new set of (necessary and sufficient) conditions for asymptotic optimality of IDA*.

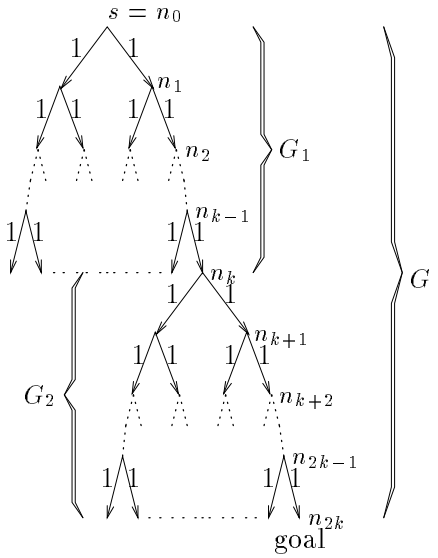

Figure 1: IDA* $\Omega(N \log N)$.

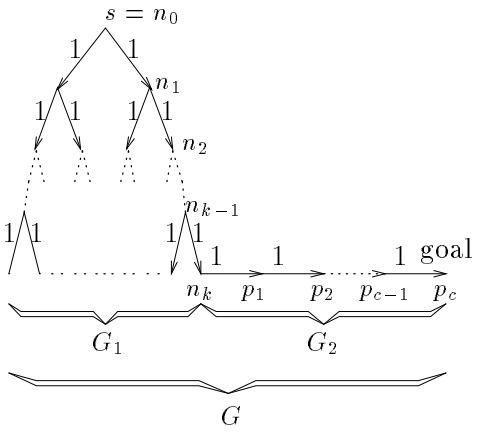

Figure 2: IDA* $O(N)$.

Asymptotic Optimality of IDA*

Let $b_1 > 1$. Then IDA*'s *active iterations* on G are the iterations $i_1^G, i_2^G, \ldots, i_{u^G}^G$ defined inductively as follows:

$i_1^G = 1$.
For $p = 2, \ldots, u$, i_p^G is the smallest integer such that $x_{\text{new}}(i^G)/x_{\text{new}}(i_{p-1}^G) \geq b_1$.

As usual, we omit the superscript G when the identity of G is obvious.

Intuitively the active iterations are the iterations in which the number of new nodes expanded by IDA* grows exponentially. We call the remaining iterations *dummy iterations*. For each i_p, let $j_{p1}, j_{p2}, \ldots, j_{pc_p}$ be the dummy iterations immediately following the active iteration i_p.

Dummy iterations can occur anywhere after the first active iteration i_1. For $q = 1, \ldots, u$, let c_q be the number of dummy iterations that occur between iterations i_q and i_{q+1}. Note that $c_q \geq 0$, and $c_1 + c_2 + \ldots + c_u = I - u$. We define $M^G = \max_q c_q$, i.e., M^G is the maximum number of adjacent dummy iterations.

The total number of node expansions by IDA* depends not only on the number of dummy iterations but also on their positions. In the following theorem we show that, keeping the total number of iterations I and the number of active iterations u fixed, the total number of node expansions by IDA* increases as the dummy iterations are moved to the right, i.e. a dummy iteration j is moved to k where $k > j$. In particular the theorem states that the total number of node expansions x_{tot} attains its maximum when all the dummy iterations appear after the last active iteration.

Theorem 2 For all positive integers N_0, u_0, I_0, let $\mathcal{G}(N_0, u_0, I_0)$ be the set of all trees G for which $N = N_0$, $u = u_0$, and $I = I_0$. Then for each N, u, I, the maximum value of x_{tot}^G over all trees $G \in \mathcal{G}(N, u, I)$ occurs in a tree G for which all dummy iterations occur after the last active iteration, i.e., $c_1 = c_2 = \ldots = c_{u-1} = 0$.

Theorem 3 provides a sufficient condition for asymptotic optimality of IDA*. It states that IDA* expands $O(N)$ nodes in every tree in which the maximum number of adjacent dummy iterations is bounded by a constant.

Theorem 3 Let $\mathcal{G} = (G_1, G_2, \ldots)$ be any arbitary sequence of trees such that $M = O(1)$ in $\mathcal{G}$. Then $x_{\text{tot}} = O(N)$ in $\mathcal{G}$.

Although the condition stated in Theorem 3 is sufficient for $O(N)$ node expansions by IDA*, it is not a necessary condition. The necessary condition is stated in Theorem 4, which can be proved using Lemma 1 (see (Mahanti *et al.*, 1992) for details). The lemma shows that if a tree G' is constructed from G in such a way that G' is identical to G except that one node n in G' has a higher f-value than in G, i.e. $f^{G'}(n) > f^G(n)$, then the total number of node expansions by IDA* on G' will be less than the number of node expansions by IDA* on G. What this means is that if a new problem instance is created from an old problem instance of IDA* by pushing a new node of iteration j to the iteration k, such that $k > j$, then x_{tot} in the new problem instance will be less than in the old problem instance. The lemma holds for the simple reason that the nodes in earlier iterations are expanded more number of times than nodes in later iterations.

Lemma 1 Let G be any tree such that $I \geq 2$, and let $1 \leq j < k \leq I$. If $x_{\text{new}}(j) = 1$, then let G' be any tree such that $I' = I - 1$ and

$$x'_{\text{new}}(i) = x_{\text{new}}(i), \qquad i = 1, \ldots, j-1;$$

$$
\begin{aligned}
x'_{\text{new}}(i) &= x_{\text{new}}(i+1), && i = j, \ldots, k-2; \\
x'_{\text{new}}(k-1) &= x_{\text{new}}(k) + 1; & & \\
x'_{\text{new}}(i) &= x_{\text{new}}(i+1), && i = k, \ldots, I-1.
\end{aligned}
$$

Otherwise, let G' be any state space such that $I' = I$ and

$$
\begin{aligned}
x'_{\text{new}}(i) &= x_{\text{new}}(i), && i = 1, \ldots, j-1; \\
x'_{\text{new}}(j) &= x_{\text{new}}(j) - 1; & & \\
x'_{\text{new}}(i) &= x_{\text{new}}(i), && i = j+1, \ldots, k-1; \\
x'_{\text{new}}(k) &= x_{\text{new}}(k) + 1; & & \\
x'_{\text{new}}(i) &= x_{\text{new}}(i+1), && i = k+1, \ldots, I.
\end{aligned}
$$

Then $x'_{\text{tot}} < x_{\text{tot}}$.

The following theorem says that IDA* achieves $O(N)$ node expansions only if the number dummy iterations after the last active iteration is bounded by a constant.

Theorem 4 Let $\mathcal{G} = (G_1, G_2, \ldots)$ be any arbitary sequence of trees. Then in $\mathcal{G}$, $x_{\text{tot}} = O(N)$ only if $c_u = O(1)$.

Limited-Memory Search on Trees

In this section, we show that in general, limited-memory best-first search algorithms can not always perform as well as A*, even on trees.

Let G be a tree, and $\mathcal{A}$ be any search algorithm used to search G. $\mathcal{A}$ *stores* a node n if during the current state of $\mathcal{A}$'s execution, $\mathcal{A}$ contains information about the identity of node n (plus possibly some other information about n). $\mathcal{A}$ *properly stores* node n if it stores not only n, but also at least one of the parents of n. $\mathcal{A}$ *properly runs* in storage $S \geq 0$ if at all times during its operation, it properly stores no more than S nodes.

Lemma 2 Let G be a b-ary tree that is complete to depth k for some $k > 0$, and $\mathcal{A}$ be a search algorithm that properly runs in storage S on G. Let d be the smallest integer such that $S \leq \frac{b^{d+1}-1}{b-1}$. If $d < k$, then $\mathcal{A}$ properly stores no more than b^d of the nodes at depth $d+1$ of G.

Let $\mathcal{A}_{bf}$ be any limited-memory best-first tree search algorithm. An algorithm is said to perform a best-first search in limited memory on tree G if for each $z > 0$, it does not expand any node of $\mathcal{V}^G(z)$ before expanding every node of $\mathcal{V}^G(z')$ at least once, for all $z' < z$. Note that IDA*, MA* (Chakrabarti *et al.*, 1989), MREC (Sen & Bagchi, 1989) are all limited-memory best-first tree search algorithms. The following theorem states that there exists no best-first tree search algorithm, which while using less than a constant fraction of the memory used by A*, can have the same worst-case time complexity as A* on all trees. Its proof uses the result of lemma 2.

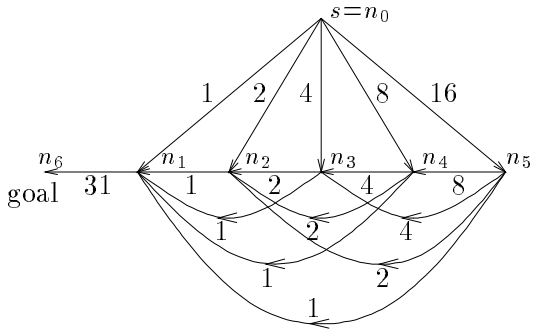

Figure 3: IDA* is $\Omega(2^{2N})$.

Theorem 5 There does not exist any best-first algorithm $\mathcal{A}_{bf}$ such that for every sequence of trees $\mathcal{G} = (G_1, G_2, \ldots)$, $\mathcal{A}_{bf}$ has $O(N)$ complexity and properly runs in $S = \frac{N}{\psi(N)}$ memory, where $\psi(N)$ is a function that is $\neq O(1)$.

IDA* on Acyclic Graphs

What happens if we run IDA* on directed acyclic graphs? For graphs with monotone heuristic, when a node n is expanded by A*, $g(n) = g^*(n)$ and A* does not expand a node more than once. Since IDA* runs in linear memory, it can not store all expanded nodes for future duplicate checking as in A*. Thus IDA* can expand a node several times due to both its limited-memory nature and unfolding of a graph into tree. It has been shown by Korf that depth-first search can expand $\Omega(2^N)$ nodes on a directed acyclic graph with N nodes (Korf, 1988). We extend this result to IDA* and show that IDA* can expand $\Omega(2^{2N})$ nodes on directed acycllic graphs with N nodes. The following example demonstrates the worst-case behavior of IDA* on directed acyclic graphs.

Example. Consider the search graph G shown in Figure 3. We can generalize the graph with $N = k+1$ non-goal nodes and one goal node. Let n_0 be the start node and n_{k+1} be the goal node. The cost structure is defined as follows:

$$
\begin{aligned}
c(n_0, n_i) &= 2^{i-1}, && 1 \leq i \leq k; \\
c(n_1, n_{k+1}) &= 2^k - 1; & & \\
c(n_i, n_{i-1}) &= 2^{i-2}, && 1 < i \leq k; \\
c(n_i, n_j) &= 2^{j-1}, && 1 \leq j < i,\ 1 < i \leq k; \\
h(n_i) &= 0, && 0 \leq i \leq k+1.
\end{aligned}
$$

It can be easily seen that the unfolded tree of G will contain nodes of all $f-$values from 0 through 2^N. Therefore the total number of node expansions will be $O(N)$ for A*, and $\Omega(2^{2N})$ for IDA*.

The following theorem gives upper bounds on the total number of node expansions by IDA* in the worst-case on trees and graphs.

Theorem 6 IDA* makes no more than N^2 node expansions on trees, and no more than 2^{2N} node expansions on acyclic graphs.

Conclusion

We have presented the following results about IDA* and related algorithms:

1. The conditions stated by Korf (Korf, 1988) are not sufficient to guarantee asymptotic optimality of IDA*; i.e., IDA* will perform badly in some of the trees on which it was thought to be asymptotically optimal.
2. The above failing is not unique to IDA*, for in general, no best-first limited-memory heuristic search algorithm can be asymptotically optimal.
3. We have presented necessary and sufficient conditions for IDA* to be asymptotically optimal. Our conditions show that IDA* is asymptotically optimal in a somewhat different range of problems than was originally believed.
4. On graphs, with a monotone heuristic IDA* can perform exponentially worse than A*. Thus, on graphs it may be preferable to use a graph search algorithm rather than using IDA*.

References

Chakrabarti, P.; Ghosh, S.; Acharya, A.; and De Sarkar, S. 1989. Heuristic Search in Restricted Memory. *AI Journal* 41 (1): 197-221.

Dechter, R.; and Pearl, J. 1985. Generalized Best-First Search Strategies and the Optimality of A*. *JACM* 32 (3): 505-536.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. Syst. Cybern.* 4 (2): 100-107

Korf, R. 1985. Depth First Iterative Deepening: An Optimal Admissible Tree Search. *AI Journal* 27 (1): 97-109.

Korf, R. 1988. Optimal Path Finding Algorithms, *Search in AI*. Edited by Kanal, L., and Kumar, V., Springer Verlag, Symbolic Computation: 200-222.

Mahanti, A., and Pal, A. 1990. A Worst-cast Time Complexity of IDA*. In Proceedings of SCCC-10 International Conference in Computer Science, 35-45. Santiago de Chile.

Mahanti. A., Ghosh, S., Nau, D. S., Pal, A. K., Kanal, L. 1992. On the Asymptotic Optimality of IDA*, Technical Report, CS-TR-xxx. Dept. of Computer Science, University of Maryland.

Patrick B. G., Almulla M. and Newborn M. M., An Upper Bound on the Complexity of Iterative-Deepening-A*.

Nilsson, N. J. 1980. *Principles of Artificial Intelligence*, Tioga Publications Co., Palo Alto, CA.

Pearl, J. 1984. *Heuristics, Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley.

Sen, A., and Bagchi A. Fast Recursive Formulations for Best-First Search That Allow Controlled Use of Memory. In Proceedings of the 11th International Joint Conference on Artificial Intelligence, 297-302.