THE OPTIMALITY OF A*

Rina Dechter
Judea Pearl

September 1987
CSD-870049

To appear in Search in AI, Springer-Verlag, 1988.

THE OPTIMALITY OF A* †

Rina Dechter and Judea Pearl

Cognitive Systems Laboratory
UCLA Computer Science Department
Los Angeles, CA. 90024-1596

ABSTRACT

This paper examines the computational optimality of A*, in the sense of never expanding a node that could be skipped by some other algorithm having access to the same heuristic information that A* uses. We define a hierarchy of four optimality types, and consider three classes of algorithms and four domains of problem instances relative to which computational performances are appraised. For each class-domain combination, we then identify the strongest type of optimality that exists and the algorithm achieving it. Our main results relate to the class of algorithms which, like A*, return optimal solutions (i.e., admissible) when all cost estimates are optimistic (i.e., $h \leq h^*$). On this class we show that A* is not optimal and that no optimal algorithm exists, but if we confine the performance tests to cases where the estimates are also consistent, then A* is indeed optimal. Additionally, we show that A* is optimal over a subset of the latter class containing all *best-first* algorithms that are guided by path-dependent evaluation functions.

1. INTRODUCTION AND PRELIMINARIES

Of all search strategies used in problem solving, one of the most popular methods of exploiting heuristic information to cut down search time is the *informed best-first* strategy. The general philosophy of this strategy is to use the heuristic information to assess the "merit" latent in every candidate search avenue exposed during the search, then continue the exploration along

† This work was supported in part by the National Science Foundation, Grant #IRI 85-01234.

the direction of highest merit. Formal descriptions of this strategy are usually given in the context of path searching problems [Nilsson, 1971., Pearl, 1984a] a formulation which represents many combinatorial problems such as routing, scheduling, speech recognition, scene analysis, and others.

Given a weighted directional graph G with a distinguished start node s and a set of goal nodes Γ, the *optimal path problem* is to find a least-cost path from s to any member of Γ where the cost of the path may, in general, be an arbitrary function of the weights assigned to the nodes and branches along that path.

By far, the most studied version of the Best-First strategies is the algorithm A* [Hart, 1968] which was developed for *additive cost measures*, i.e, where the cost of a path is defined as the sum of the costs of its arcs. To match this cost measure, A* employs an additive evaluation function $f(n) = g(n) + h(n)$, where $g(n)$ is the cost of the currently evaluated path from s to n and h is a heuristic estimate of the cost of the path remaining between n and some goal node. A* constructs a tree T of selected paths of G using the elementary operation of *node-expansion*, i.e. generating all successors of a given node. Starting with s, A* selects for expansion that leaf node of T which has the lowest f value, and only maintains the lowest f-path to any given node. The search halts as soon as a node selected for expansion is found to satisfy the goal conditions. It is known that if $h(n)$ is a lower bound to the cost of any continuation path from n to Γ, then A* is *admissible*, that is, it is guaranteed to find the optimal path. A* is described in Figure 1.

Algorithm A*[1]

1. Put the start node, s, on a list called OPEN of unexpanded nodes.

2. IF OPEN is empty, exit with failure; no solution exists.

3. Remove from OPEN a node, n, at which $f=g+h$ is minimum (break ties arbitrarily, but in favor of a goal node) and place it on a list called CLOSED to be used for expanded nodes.

4. If n is a goal node, exit successfully with the solution obtained by tracing back the path along the pointers from n to s, (pointers are assigned in Steps 5 and 6).

5. Expand node n, generating all its successors with pointers back to n.

6. For every successor n' of n:

   a. Calculate $f(n')$

   b. If n' was neither in OPEN nor in CLOSED, then add it to OPEN. Assign the newly computed $f(n')$ to node n'.

   c. If n' already resided in OPEN or CLOSED, compare the newly computed $f(n')$ with that previously assigned to n'. If the new value is lower, substitute it for the old (n' now points back to n instead of to its predecessor). If the matching node n' resided in CLOSED, move it back to OPEN.

7. Go to (2).

Figure 1: Algorithm A*

A Best-First algorithm uses an evaluation function f which may be any function of the path parameters, not necessarily the additive combination of g and h. Substituting this function in the description of A* defines the general class of Best-First algorithms to which (among others) we compare A*. For an elaborate discussion of best-first strategies see [Dechter 1985].

[1] Our definition of A* is identical to that of [Nilsson 1971.] and is at variance with [Nilsson 1980]. The latter regards the requirement $h \leq h^*$ as part of the definition of A*, otherwise the algorithm is called A. We found it more convenient to follow the tradition of identifying an algorithm by how it processes input information rather than by the type of information that it may encounter. Accordingly, we assign the symbol A* to any best-first algorithm which uses the additive combination $f = g+h$, placing no restriction on h, in line with the more recent literature [Barr 1981, Bagchi 1983, Pearl 1984b]

In this paper, our aim is to examine under what conditions A* (employing $f=g+h$) is computationally optimal over other search algorithms which are provided with the same heuristic information h and are guaranteed to find solutions comparable to those found by A*. As a measure of complexity we will count the number of distinct nodes expanded by the algorithm. Since a given node can be expanded several times, the measure chosen is simpler to analyze and in many practical cases (see conclusions) this measure coincides with the number of expansions.

Notation:

G - directed locally finite graph, $G=(V,E)$

C^* - The cost of the cheapest solution path.

$C(.)$ - the cost function defined over all solution paths

Γ - a set of goal nodes, $\Gamma \subseteq V$

$P_{n_i-n_j}$ - A path in G between node n_i and n_j.

P^s - a *solution path*, i.e., a path in G from s to some goal node $\gamma \in \Gamma$

$c(n,n')$ - cost of an arc between n and n', $c(n,n') \geq \delta > 0$, where δ is a constant.

$f(.)$ - evaluation function defined over partial paths, i.e., to each node n along a given path $P=s_1,n_1,n_2,\ldots,n$ we assign the value $f_P(n)$ which is a shorthand notation for $f(s,n_1,n_2,\ldots,n)$.

$g(n)$ - The sum of the branch costs along the current path of pointers from n to s.

$g^*(n)$ - The cost of the cheapest path going from s to n

$g_P(n)$ - The sum of the branch costs along path P from s to n.

$h(n)$ - A cost estimate of the cheapest path remaining between n and Γ.

$h^*(n)$ - The cost of the cheapest path going from n to Γ

$k(n,n')$ - cost of the cheapest path between n and n'

s - start node

T - A subtree of G containing all the arcs to which pointers are currently assigned.

2. PREVIOUS WORKS AND THE NOTION OF EQUALLY-INFORMED

The *optimality* of A*, in the sense of *computational efficiency*, has been a subject of some confusion. The well-known property of A* which predicts that decreasing errors h^*-h can only improve its performance (result 6 in [Nilsson 1980]) has often been interpreted to reflect some supremacy of A*

over other search algorithms of equal information. Consequently, several authors have assumed that A*'s optimality is an established fact (e.g [Nilsson 1971., Mero 1984].). In fact, all this property says is that some A* algorithms are better than other A* algorithms depending on the heuristics which guide them. It does not indicate whether the additive rule $f=g+h$ is the best way of combining g and h, neither does it assure us that expansion policies based only on g and h can do as well as more sophisticated policies that use the entire information gathered by the search. These two conjectures will be examined, and will be given a qualified confirmation.

The first attempt to prove the optimality of A* was carried out by Hart, Nilsson and Raphael [Hart 1968] and is summarized in [Nilsson 1971.]. Basically, Hart et al. argue that if some admissible algorithm B fails to expand a node n expanded by A*, then B must have known that any path to a goal constrained to go through node n is nonoptimal. A*, by comparison, had no way of realizing this fact because when n was chosen for expansion it satisfied $g(n)+h(n) \leq C^*$, clearly advertizing its promise to deliver an optimal solution path. Thus, the argument goes, B must have obtained extra information from some external source, unavailable to A* (perhaps by computing a higher value for $h(n)$), and this disqualifies B from being an "equally informed", fair competitor, to A*.

The weakness of this argument is that it fails to account for two legitimate ways in which B can decide to refrain from expanding n based on information perfectly accessible to A*. First, B may examine the properties of previously exposed portions of the graph and infer that n actually deserves a much higher estimate than $h(n)$. A*, on the other hand, although it has the same information available to it in CLOSED, cannot put it into use because it is restricted to take the estimate $h(n)$ *at face value* and only judge nodes by the score $g(n)+h(n)$. Second, B may also gather information while exploring sections of the graph unvisited by A*, and this should not render B an unfair, "more informed" competitor to A* because in principle A* too had an opportunity to visit those sections of the graph. Later (see Figure 4) we demonstrate the existence of an algorithm B which manages to outperform A* using this kind of information.

Gelperin [Gelperin 1977] has correctly pointed out that in any discussion of the optimality of A* one should also consider algorithms which adjust their h in accordance with the information gathered during the search. His analysis, unfortunately, falls short of considering the entirety of this extended class, having to follow an over-restrictive definition of *equally-*

informed. Gelperin's interpretation of "an algorithm B is *never more informed* than A*", instead of just restricting B from using information inaccessible to A, actually forbids B from processing common information in a better way than A does. For example, if B is a best-first algorithm guided by the evaluation function f_B, then in order to qualify for Gelperin's definition of "never more informed than A*," B is forbidden from ever assigning to a node n a value $f_B(n)$ higher than $g(n)+h(n)$, even if the information gathered along the path to n justifies such an assignment.

In our analysis we will use the natural definition of "equally informed," allowing the algorithms compared to have access to the same heuristic information while placing no restriction on the way they use it. Accordingly, we assume that an arbitrary heuristic function $h(n)$ is assigned to the nodes of G and that the value $h(n)$ is made available to each algorithm that chooses to generate node n. This amounts to viewing $h(n)$ as part of the parameters that specify problem-instances and correspondingly, we shall represent each problem instance by the quadruple $I = (G, s, \Gamma, h)$.

We will demand, however, that A* only be compared to algorithms that return optimal solutions in those problem instances where their computational performances are to be appraised. In particular, if our problem space contains only cases where $h(n) \leq h^*(n)$ for every n in G, we will only consider algorithms which, like A*, return least-cost solutions, in such cases. The class of algorithms answering this conditional admissibility requirement will simply be called *admissible* and will be denoted by $\underline{A}_{ad}$. From this general class of algorithms we will later examine two subclasses $\underline{A}_{gc}$ and $\underline{A}_{bf}$. $\underline{A}_{gc}$ denotes the class of algorithms which are *globally compatible* with A*, i.e., they return optimal solutions whenever A* does, even in cases where $h > h^*$. $\underline{A}_{bf}$ stands for the class of admissible Best-First algorithms, i.e., those which, like A*, conduct their search in a best-first manner, being guided by any path-dependent evaluation function.

Additionally, we will assume that each algorithm compared to A* uses the primitive computational step of *node expansion*, that it only expands nodes which were generated before, and that it begins the expansion process at the start node s. This excludes, for instance, bi-directional searches [Pohl 1971] or algorithms which simultaneously grow search trees from several "seed nodes" across G.

3. NOMENCLATURE AND A HIERARCHY OF OPTIMALITY RELATION

Our notion of *optimality* is based on the usual requirement of *Dominance*.

Definition: Algorithm A is said to *dominate* algorithm B relative to a set $\underline{I}$ of problem instances iff in every instance $I \in \underline{I}$, the set of nodes expanded by A is a subset of the set of nodes expanded by B. A *strictly dominates* B iff A dominates B and B does not dominate A, i.e., there is at least one instance where A skips a node which B expands, and no instance where the opposite occurs.

This definition is rather stringent because it requires that A establishes its superiority over B under two difficult tests:

1. expanding a *subset* of nodes rather than a *smaller number* of nodes
2. outperform B in *every* problem instance rather than the *majority* of instances

Unfortunately, there is no easy way of loosening any of these requirements without invoking statistical assumptions regarding the relative likelihood of instances in $\underline{I}$. In the absence of an adequate statistical model, requiring dominance remains the only practical way of guaranteeing that A expands *fewer* nodes than B, because if in some problem instance we would allow B to skip even one node that is expanded by A, one could immediately present an infinite set of instances where B *grossly* outperforms A. (This is normally done by appending to the node skipped a variety of trees with negligible costs and very low h).

Adhering to the concept of dominance, the strong definition of optimality proclaims algorithm A *optimal* over a class $\underline{A}$ of algorithms iff A dominates every member of $\underline{A}$. Here the combined multiplicity of $\underline{A}$ and $\underline{I}$ also permits weaker definitions, for example, we may proclaim A *weakly optimal* over $\underline{A}$ if no member of $\underline{A}$ strictly dominates A. The spectrum of optimality conditions becomes even richer when we examine A*, which stands for not just one but a whole family of algorithms, each defined by the tie-breaking-rule chosen. We chose to classify this spectrum into the following four types (in a hierarchy of decreasing strength):

Type 0: A* is said to be 0-optimal over $\underline{A}$ relative to $\underline{I}$ iff in every problem

instance

$I \in \underline{I}$ every tie-breaking-rule in A* expands a subset of the nodes expanded by any member of $\underline{A}$. (In other words, every tie-breaking-rule dominates all members of $\underline{A}$.)

Type 1: A* is said to be 1-optimal over $\underline{A}$ relative to $\underline{I}$ iff in every problem instance $I \in \underline{I}$ there exists at least one tie-breaking-rule which expands a subset of the set of nodes expanded by any member of $\underline{A}$.

Type 2: A* is said to be 2-optimal over $\underline{A}$ relative to $\underline{I}$ iff there exists no problem instance $I \in \underline{I}$ where some member of $\underline{A}$ expands a proper subset of the set of nodes which are expanded by some tie-breaking-rule in A*.

Type 3: A* is said to be 3-optimal over $\underline{A}$ relative to $\underline{I}$ iff the following holds: if there exists a problem instance $I_1 \in \underline{I}$ where some algorithm $B \in \underline{A}$ skips a node expanded by some tie-breaking-rule in A*, then there must also exist some problem instance $I_2 \in \underline{I}$ where that tie-breaking-rule skips a node expanded by B. (In other words, no tie-breaking-rule in A* is strictly dominated by some member of $\underline{A}$.)

Type-1 describes the notion of optimality most commonly used in the literature, and it is sometimes called "optimal up to a choice of a tie-breaking-rule". Note that these four definitions are applicable to any class of algorithms, $\underline{B}$, contending to be optimal over $\underline{A}$; we need only replace the words "tie-breaking-rule in A*" by the words "member of $\underline{B}$". If $\underline{B}$ turns out to be a singleton, then type-0 and type-1 collapse to strong optimality. Type-3 will collapse into type-2 if we insist that I_1 be identical to I_2. Note also that the hierarchy is not strict, since type-1 does not necessarily imply type-2.

We are now ready to introduce the four domains of problem instances over which the optimality of A* is to be examined. The first two relate to the admissibility and consistency of $h(n)$.

Definition: A heuristic function $h(n)$ is said to be *admissible* on (G, Γ) iff $h(n) \leq h^*(n)$ for every $n \in G$

Definition: A heuristic function $h(n)$ is said to be *consistent* (or monotone) on G iff for any pair of nodes, n' and n, the triangle inequality holds:

$$h(n') \leq k(n',n) + h(n) \tag{1}$$

Corresponding to these two properties we define the following sets of problem instances:

$$\underline{I}_{AD} = \left\{ (G,s,\Gamma,h) \mid h \leq h^* \text{ on } (G,\Gamma) \right\} \tag{2}$$

$$\underline{I}_{CON} = \left\{ (G,s,\Gamma,h) \mid h \text{ is consistent on } G \right\} \tag{3}$$

Clearly, consistency implies admissibility [Pearl 1984b] but not vice versa, therefore, $\underline{I}_{CON} \subseteq I_{AD}$

A special and important subset of $\underline{I}_{AD}$ (and $\underline{I}_{CON}$), called *non-pathological instances*, are those instances for which there exists at least one optimal solution path along which h is not fully informed, that is, $h<h^*$ for every non-goal node on that path. The non-pathological subsets of $\underline{I}_{AD}$ and $\underline{I}_{CON}$ will be denoted by $\underline{I}^{-}_{AD}$ and $\underline{I}^{-}_{CON}$, respectively. The term ''pathological'' should not connote ''rareness''; many real world problems contain such situations when the node is very close to the goal. However, in practice this occurs only when the node is very close to the goal.

It is known that if $h \leq h^*$, then A* expands every node reachable from s by a strictly C^*-bounded path, regardless of the tie-breaking rule used. The set of nodes with this property will be referred to as *surely expanded* by A*. In general, for an arbitrary constant d and an arbitrary evaluation function f over (G, s, Γ, h), we let $\underline{N}^{d}_{f}$ denote the set of all nodes reachable by a path from s whose f values are strictly d-bounded. For example, $\underline{N}^{C^*}_{g+h}$ is a set of nodes surely expanded by A* in some instance of $\underline{I}_{AD}$.

The importance of non-pathological instances lies in the fact that in such instances the set of nodes surely expanded by A* are indeed *all* the nodes expanded by it. Therefore for these instances any claim regarding the set of nodes surely expanded by A* can be translated to "the set of all the nodes" expanded by A*. This is not the case, however, for pathological instances in $\underline{I}_{AD}$; $\underline{N}^{C^*}_{g+h}$ is often a proper subset of the set of nodes actually expanded by A*. If h is consistent, then the two sets differ only by nodes for which $h(n) = C^* - g^*(n)$ [Pearl 1984b]. However, in cases where h is inconsistent, the difference may be very substantial; each node n for which

$h(n) = C^* - g^*(n)$ may have many descendents assigned lower h values (satisfying $h+g<C^*$) and these descendents may be expanded by every tie-breaking-rule of A* even though they do not belong to $\underline{N}^{C^*}_{g+h}$.

In the following section we present several theorems regarding the behavior of competing classes of algorithms relative to the set $\underline{N}^{C^*}_{g+h}$ of nodes surely expanded by A*, and will interpret these theorems as claims about the type of optimality that A* enjoys over the competing classes. Moreover, for any given pair $(\underline{A}, \underline{I})$ where $\underline{A}$ is a class of algorithms drawn from $\{\underline{A}_{ad}, \underline{A}_{bf}, \underline{A}_{gc}\}$ and $\underline{I}$ is a domain of problem instances from $\{\underline{I}_{AD}, \underline{I}_{\bar{A}D}, \underline{I}_{CON}, \underline{I}_{\bar{C}ON}\}$, we will determine the strongest type of optimality that can be established over $\underline{A}$ relative to $\underline{I}$, and will identify the algorithm that achieves this optimality. The relationships between these classes of algorithms and problem domains are shown in Figure 2. The algorithm A** is an improvement over A* discussed in the Appendix.

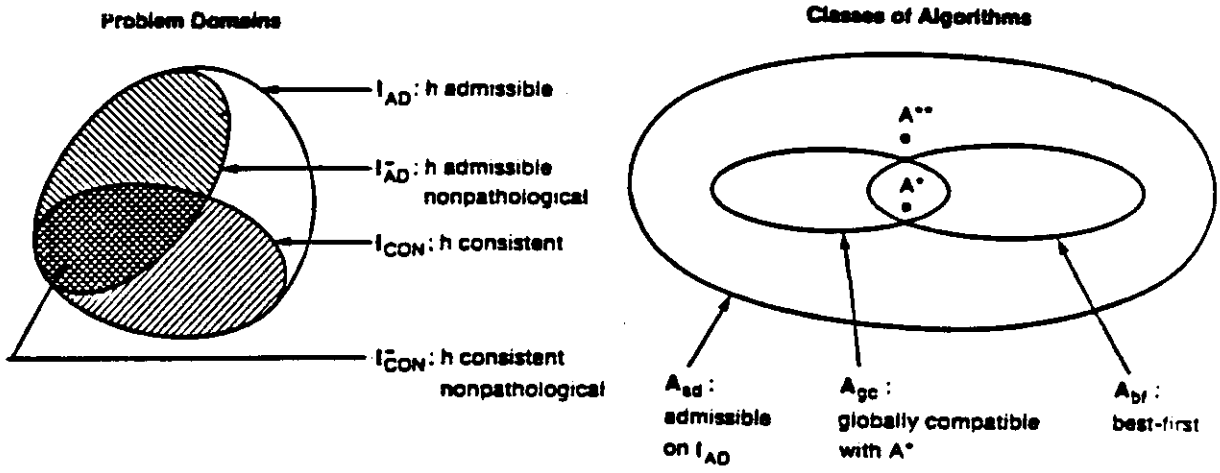

Figure 2 - The classes of algorithm and the problem instances for which the optimality of A* is examined.

4. WHERE AND HOW IS A* OPTIMAL?

4.1 Optimality Over Admissible Algorithms, $\underline{A}_{ad}$

Theorem 1:

Any algorithm that is admissible on $\underline{I}_{AD}$ will expand, in every instance $I \in \underline{I}_{CON}$, all nodes surely expanded by A*.

Proof:

Let $I=(G,s,\Gamma,h)$ be some problem instance in $\underline{I}_{CON}$ and assume that n is surely expanded by A*, i.e., $n \in \underline{N}^{C^*}_{g+h}$. Therefore, there exists a path P_{s-n} such that

$$\forall n' \in P_{s-n} \ , \ g(n') + h(n') < C^* \tag{4}$$

Let B be an algorithm compatible with A*, namely halting with cost C^* in I.

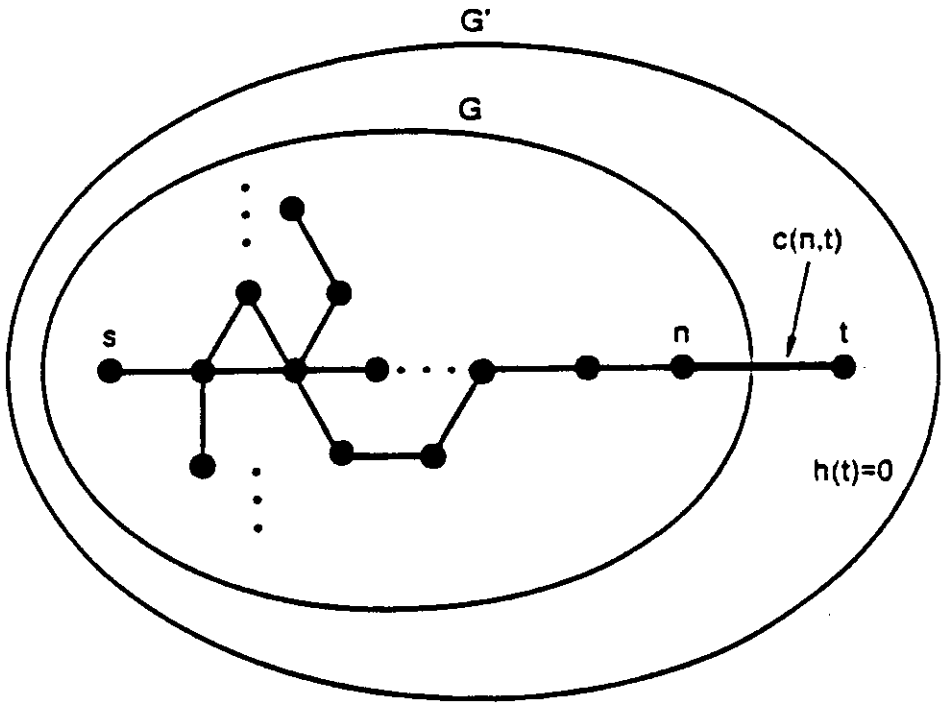

Figure 3 - The graph G' represents a new problem instance constructed by appending to n a branch leading to a new goal node t.

Assume that B does not expand n. We now create a new graph G' (see figure 3) by adding to G a goal node t with $h(t)=0$ and an edge from n to t with non-negative cost $c=h(n)+\Delta$, where

$$\Delta = 1/2(C^* - D) > 0 \tag{5}$$

and

$$D = \max\left\{ f(n') \mid n' \in \underline{N}^{C^*}_{g+h} \right\} \tag{6}$$

This construction creates a new solution path P^* with cost at most $C^*-\Delta$ and,

simultaneously, (due to h's consistency on I) retains the consistency (and admissibility) of h on the new instance I'. To establish the consistency of h in I' we note that since we kept the h values of all nodes in G unchanged, consistency will continue to hold between any pair of nodes previously in G. It remains to verify consistency on pairs involving the new goal node t, which amounts to establishing the inequality $h(n') \leq k(n',t)$ for every node n' in G. Now, if at some node n' we have $h(n') > k(n',t)$ then we should also have:

$$h(n') > k(n',n) + c = k(n',n) + h(n) + \Delta \tag{7}$$

in violation of h's consistency on I. Thus, the new instance is also in $\underline{I}_{CON}$.

In searching G', algorithm A* will find the extended path $P*$ costing C^*-Δ, because:

$$f(t) = g(n)+c = f(n)+\Delta \leq D + \Delta = C^* - \Delta < C^* \tag{8}$$

and so, t is reachable from s by a path bounded by C^*-Δ which ensures its selection. Algorithm B, on the other hand, if it avoids expanding n, must behave the same as in problem instance I, halting with cost C^* which is higher than that found by A*. This contradicts the supposition that B is both admissible on I and avoids the expansion of node n.

□

The implications of Theorem 1 relative to the optimality of A* are rather strong. In non-pathological cases $I \in \underline{I}^{-}_{CON}$ A* never expands a node outside $\underline{N}^{C^*}_{g+h}$ and, therefore, Theorem 1 establishes the 0-optimality of A* over all admissible algorithms relative to $\underline{I}^{-}_{CON}$. In pathological cases of $\underline{I}_{CON}$ there may also be nodes satisfying $f(n)=C^*$ that some tie-breaking-rule in A* expands and, since these nodes are defined to be outside $\underline{N}^{C^*}_{g+h}$, they may be avoided by some algorithm $B \in \underline{A}_{ad}$, thus destroying the 0-optimality of A* relative to all $\underline{I}_{CON}$. However, since there is always a tie-breaking-rule in A* which, in addition to $\underline{N}^{C^*}_{g+h}$, expands only nodes along one optimal path, Theorem 1 also establishes the 1-optimality of A* relative the entire $\underline{I}_{CON}$ domain. Stronger yet, the only nodes that A* expands outside $\underline{N}^{C^*}_{g+h}$ are those satisfying $f(n) = C^*$, and since this equality is not likely to occur in many nodes of the graph, we may interpret Theorem 1 to endow A* with "almost" 0-optimality (over all admissible algorithms) relative to $\underline{I}_{CON}$.

The proof of Theorem 1 makes it tempting to conjecture that A* retains the same type of optimality relative to cases where h is admissible but not necessarily consistent. In fact, the original argument of Hart, Nilsson and Raphael [Hart 1968] that no admissible algorithm equally informed to A* can ever avoid a node expanded by A* (see Section 2), amounts to claiming that A* is at least 1-optimal relative to $\underline{I}_{AD}$. Similar claims are made by Mero [Mero 1984] and are suggested by the theorems of [Gelperin 1977].

Unfortunately, Theorem 1 does not lend itself to such extension; if h is admissible but not consistent, then after adding the extra goal node t to G (as in Figure 3) we can no longer guarantee that h will remain admissible on the new instance created. Furthermore, we can actually construct an algorithm that is admissible on $\underline{I}_{AD}$ and yet, in some problem instances, it will grossly outperform every tie-breaking-rule in A*. Consider an algorithm B guided by the following search policy: Conduct an exhaustive right-to-left depth-first search but refrain from expanding one distinguished node n, e.g., the leftmost son of s. By the time this search is completed, examine n to see if it has the potential of sprouting a solution path cheaper than all those discovered so far. If it has, expand it and continue the search exhaustively. Otherwise,[3] return the cheapest solution at hand. B is clearly admissible; it cannot miss an optimal path because it would only avoid expanding n when it has sufficient information to justify this action, but otherwise will leave no stone unturned. Yet, in the graph of Figure 4a, B will avoid expanding many nodes which are surely expanded by A*. A* will expand node J_1 immediately after s ($f(J_1)$=4) and subsequently will also expand many nodes in the subtree rooted at J_1. B, on the other hand, will expand J_3, then select for expansion the goal node γ, continue to expand J_2 and at this point will halt without expanding node J_1. Relying on the admissibility of h, B can infer that the estimate $h(J_1)$=1 is overly optimistic and should be at least equal to $h(J_2)$−1=19, thus precluding J_1 from lying on a solution path cheaper than the path (s, J_3, γ) at hand.

Granted that A* is not 1-optimal over all admissible algorithms relative to $\underline{I}_{AD}$, the question arises if a 1-optimal algorithm exists altogether. Clearly, if a 1-optimal algorithm exists, it would have to be better than A* in the sense of skipping in some problem instances, at least one node surely expanded by A* while never expanding a node which is surely skipped by A*. Note that

---

[3] A simple valid test for skipping a node in $\underline{I}_{AD}$ is that $\max\{g(n')+h(n') \mid n'\}$ on some path P from s to n be larger than the cost of the cheapest solution at hand.

algorithm B above could not be such an optimal algorithm because in return for skipping node J_1 in Figure 2.6a, it had to pay the price of expanding J_2, and J_2 will not be expanded by A* regardless of the tie-breaking-rule invoked. If we could show that this "node tradeoff" pattern must hold for every admissible algorithm and on every instance of $\underline{I}_{AD}$, then we would have to conclude both that no 1-optimal algorithm exists and that A* is 2-optimal relative to this domain. Theorem 2 accomplishes this task relative to $\underline{I}_{AD}^{-}$.

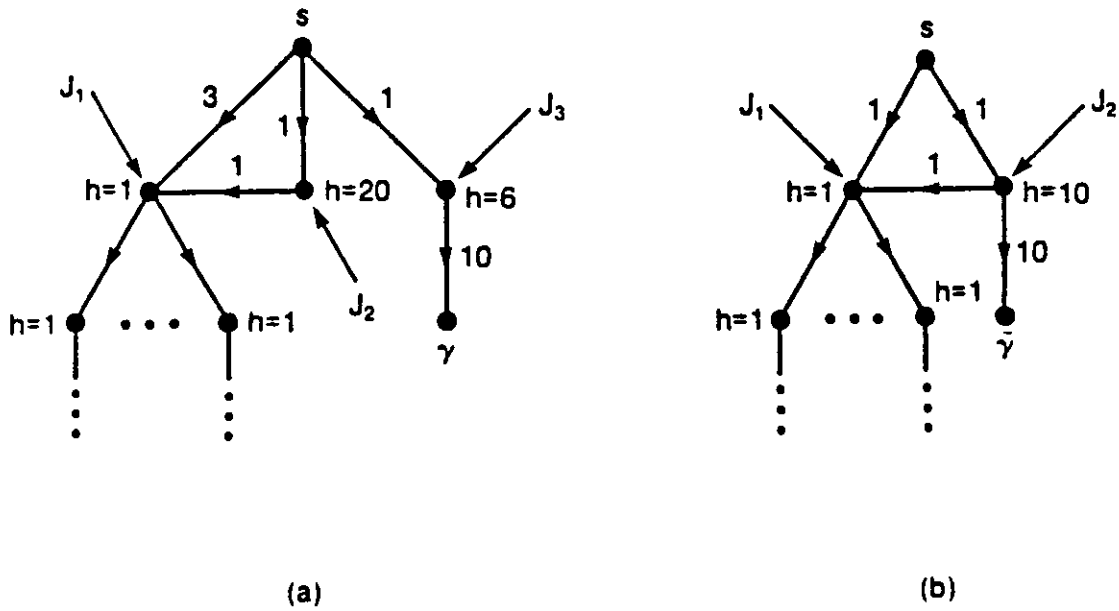

Figure 4 - Graphs demonstrating that A* is not optimal.

Theorem 2:

If an admissible algorithm B does not expand a node which is surely expanded by A* in some problem instance where h is admissible and non-pathological, then in that very problem instance B must expand a node which is avoided by every tie-breaking-rule in A*.

Proof:

Assume the contrary, i.e., there is an instance $I=(G,s,\Gamma,h) \in \underline{I}_{AD}^{-}$ such that a node n which is surely expanded by A* is avoided by B and, at the same time, B expands no node which is avoided by A*, we shall show that this assumption implies the existence of another instance $I' \in \underline{I}_{AD}$ where B will not find an optimal solution. I' is constructed by taking the graph G_e exposed by the run of A* (including nodes in OPEN) and appending to it another edge

(n,t) to a new goal node t, with cost $c(n,t)=D-k_e(s,n)$ where

$$D = \max\left\{ f(n') \mid n' \in \underline{N}^{C^*}_{g+h} \right\} \tag{9}$$

and $k_e(n',n)$ is the cost of the cheapest path from n' to n in G_e.

Since G contains an optimal path $P^*_{s\_}\gamma$ along which $h(n')<h^*(n')$ (with the exception of γ and possibly s), we know that because ties are broken in favor of goal nodes A* will halt without ever expanding a node having $f(n)=C^*$. Therefore, every nonterminal node in G_e must satisfy the strict inequality $g(n)+h(n)<C^*$.

We shall first prove that I' is in $\underline{I}_{AD}$, i.e., that $h(n') \le h^*_{I'}(n')$ for every node n' in G_e. This inequality certainly holds for n' such that $g(n')+h(n') \ge C^*$ because all such nodes were left unexpanded by A* and hence appear as terminal nodes in G_e for which $h^*_{I'}(n')=\infty$ (with the exception of γ, for which $h(\gamma)=h^*_{I'}(\gamma)=0$). It remains, therefore, to verify the inequality for nodes n' in $\underline{N}^{C^*}_{g+h}$ for which we have $g(n')+h(n') \le D$. Assume the contrary, that for some $n' \in \underline{N}^{C^*}_{g+h}$ we have $h(n')>h^*_{I'}(n')$. This implies

$$\begin{aligned} h(n') &> k_e(n',n) + c(n,t) \\ &= k_e(n',n) + D - k_e(s,n) \\ &\ge k_e(n',n) + k_e(s,n') + h(n') - k_e(s,n) \end{aligned} \tag{10}$$

or

$$k_e(s,n) > k_e(n',n) + k_e(s,n') \tag{11}$$

in violation of the triangle inequality for cheapest paths in G_e. Hence, I' is in $\underline{I}_{AD}$.

Assume now that algorithm B does not generate any node outside G_e. If B has avoided expanding n in I, it should also avoid expanding n in I'; all decisions must be the same in both cases since the sequence of nodes generated (including those in OPEN) is the same. On the other hand, the cheapest path in I' now goes from s to n to t', having the cost $D<C^*$, and will be missed by B. This violates the admissibility of B on an instance in $\underline{I}_{AD}$ and proves that B could not possibly avoid the expansion of n without generating at least one node outside G_e. Hence, B must expand at least one node avoided by A* in this specific run. □

Theorem 2 has two implications. On one hand it conveys the discomforting fact that neither A* nor any other algorithm is 1-optimal over those guaranteed to find an optimal solution when given $h \leq h^*$. On the other hand, Theorem 2 endows A* with some optimality property, albeit weaker than hoped; the only way to gain one node from A* is to relinquish another. Not every algorithm enjoys such strength. These implications are summarized in the following Corollary.

Corollary 1: No algorithm can be 1-optimal over all admissible algorithms relative to $\underline{I}_{AD}$, but A* is 2-optimal over this class relative to $\underline{I}_{AD}^{-}$.

The fact that Theorem 2 had to be limited to non-pathological instances is explained by Figure 4b, showing an exception to the node-tradeoff rule on a pathological instance. Algorithm B does not expand a node (J_1) which must be expanded by A* and yet, B does not expand any node which A* may skip. This example implies that A* is not 2-optimal relative to the entire $\underline{I}_{AD}$ domain and, again, this begs the questions whether there exists a 2-optimal algorithm altogether, and whether A* is at least 3-optimal relative to this domain.

The answer to both questions is, again, negative; another algorithm that we shall call A**, turns out both, to strictly dominate A* and to meet the requirements for type-3 optimality relative to $\underline{I}_{AD}$. A** conducts the search in a manner similar to A*, with one exception; instead of $f(n)=g(n)+h(n)$, A** uses the evaluation function:

$$f(n) = \max\left\{ g(n') + h(n') \mid n' \text{ on the current path to } n \right\} \qquad (12)$$

This, in effect, is equivalent to raising the value of $h(n)$ to a level where it becomes consistent with the h's assigned to the ancestors of n [Mero 1984]. A** chooses for expansion the nodes with the lowest f value in OPEN (breaking ties arbitrarily but in favor of goal nodes) and adjusts pointers along the path having the lowest g value. In figure 4a, for example, if A** ever expands node J_2 then its son J_1 will immediately be assigned the value $f(J_1) = 21$ and its pointer be directed toward J_2.

It is possible to show (see Appendix) that A** is admissible and that in non-pathological cases A** expands the same set of nodes as does A*, namely the surely expanded nodes in $\underline{N}^{C^*}_{g+h}$. In pathological cases, however, there exist tie-breaking-rules in A** that strictly dominate every tie-breaking-rule in A*. This immediately precludes A* from being 3-optimal relative to $\underline{I}_{AD}$ and nominates A** for that title.

Theorem 3:
Let a** be some tie-breaking-rule in A** and B an arbitrary algorithm, admissible relative to $\underline{I}_{AD}$. If in some problem instance $I_1 \in \underline{I}_{AD}$, B skips a node expanded by a** then there exists another instance $I_2 \in \underline{I}_{AD}$ where B expands a node skipped by a**.

Proof:
Let

$$S_A = n_1, n_2, \ldots n_k, J \ \cdots \tag{13}$$

and

$$S_B = n_1, n_2, \ldots n_k, K \ \cdots \tag{14}$$

be the sequences of nodes expanded by a** and B, respectively, in problem instance $I_1 \in \underline{I}_{AD}$, i.e., K is the first node in which the sequence S_B deviates from S_A. Consider G_e, the explored portion of the graph just before a** expands node J. That same graph is also exposed by B before it decides to expand K instead. Now construct a new problem instance I_2 consisting of G_e appended by a branch (J,t) with cost $c(J,t) = f(J) - g(J)$, where t is a goal node and $f(J)$ and $g(J)$ are the values that a** computed for J before its expansion. I_2 is also in $\underline{I}_{AD}$ because $h(t)=0$ and $C(J,t)$ are consistent with $h(J)$ and with the h's of all ancestors of J in G_e. For if some ancestor n_i of J satisfies $h(n_i) > h^*(n_i)$ we will obtain a contradiction:

$$\begin{aligned} g(n_i) + h(n_i) &> g(n_i) + h^*(n_i) \\ &= g(n_i) + k_e(n_i, J) + c(J,t) \\ &\geq g(J) + c(J,t) \\ &= f(J) \equiv \max_j [g(n_j) + h(n_j)] \end{aligned} \tag{15}$$

Moreover, a** will expand in I_2 the same sequence of nodes as it did in I_1, until J is expanded, at which time t enters OPEN with $f(t) = \max[g(J) + c(J,t), f(J)] = f(J)$. Now, since J was chosen for

expansion by virtue of its lowest f value in OPEN, and since a** always breaks up ties in favor of a goal node, the next and final node that a** expands must be t. Now consider B. The sequence of nodes it expands in I_2 is identical to that traced in I_1 because, by avoiding node J, B has no way of knowing that a goal node has been appended to G_e. Thus, B will expand K (and perhaps more nodes on OPEN), a node skipped by a**.

□

Note that the special evaluation function used by A** $f(n) = \max\{g(n') + h(n') \mid n' \text{ on } P_{s-n}\}$ was necessary to ensure that the new instance, I_2, remains in I_{AD}. The proof cannot be carried out for A* because the evaluation function $f(n)=g(n)+h(n)$ results in $c(J,t) = h(J)$, which may lead to violation of $h(n_i) \leq h^*(n_i)$ for some ancestor of J.

Theorem 3, together with the fact that its proof makes no use of the assumption that B is admissible, gives rise to the following conclusion:

Corollary 2: A** is 3-optimal over all algorithms relative to $\underline{I}_{AD}$.

Theorem 3 also implies that there is no 2-optimal algorithm over $\underline{A}_{ad}$ relative to $\underline{I}_{AD}$. From the 3-optimality of A** we conclude that every 2-optimal algorithm, if such exists, must be a member of the A** family, but figure 4b demonstrates an instance of $\underline{I}_{AD}$ where another algorithm (B) only expands a proper subset of the nodes expanded by every member of A**. This establishes the desired conclusion:

Corollary 3: There is no 2-optimal algorithm over $\underline{A}_{ad}$ relative to $\underline{I}_{AD}$

4.2 Optimality Over Globally Compatible Algorithms, $\underline{A}_{gc}$

So far our analysis was restricted to algorithms in $\underline{A}_{ad}$, i.e., those which return optimal solutions if $h(n) \leq h^*(n)$ for all n, but which may return arbitrarily poor solutions if there are some n for which $h(n)>h^*(n)$. In situations where the solution costs are crucial and where h may occasionally overestimate h^* it is important to limit the choice of algorithms to those which return reasonably good solutions even when $h>h^*$. A*, for example, provides such guarantees; the costs of the solutions returned by A* do not exceed $C^* + \Delta$ where Δ is the highest error $h^*(n)-h(n)$ over all nodes in the graph [Harris 1974] and, moreover, A* still returns optimal solutions in many problem

instances, i.e., whenever Δ is zero along some optimal path. This motivates the definition of $\underline{A}_{gc}$, the class of algorithms *globally compatible* with A*, namely, they return optimal solutions in *every* problem instance where A* returns such solution.

Since $\underline{A}_{gc}$ is a subset of $\underline{A}_{ad}$, we should expect A* to hold a stronger optimality status over $\underline{A}_{gc}$, at least relative to instances drawn from $\underline{I}_{AD}$. The following Theorem confirms this expectation.

Theorem 4:
Any algorithm that is globally compatible with A* will expand, in problem instances where h is admissible, all nodes surely expanded by A*.

Proof:
Let $I = (G, s, \Gamma, h)$ be some problem instance in $\underline{I}_{AD}$ and let node n be surely expanded by A*, i.e., there exists a path P_{s-n} such that

$$g(n) + h(n) < C^* \quad \text{for all } n' \in P_{s-n} \tag{16}$$

Let $D = \max\left\{ f(n') \mid n' \in P_{s-n} \right\}$ and assume that some algorithm $B \in \underline{A}_{gc}$ fails to expand n. Since $I \in \underline{I}_{AD}$, both A* and B will return cost C^*, while $D < C^*$.

We now create a new graph G', as in figure 3, by adding to G a goal node t' with $h(t') = 0$ and an edge from n to t' with non-negative cost $D - g(P_{s-n})$. Denote the extended path $P_{s-n-t'}$ by P^*, and let $I' = (G', s, \Gamma \cup t', h)$ be a new instance in the algorithms' domain. Although h may no longer be admissible on I', the construction of I' guarantees that $f(n') \leq D$ if $n' \in P^*$. Therefore, there will always be an OPEN node with $f(n) \leq D$ and, algorithm A* searching G' will find an optimal solution path with cost $C_t \leq D$. Algorithm B, however, will search I' in exactly the same way it searched I; the only way B can reveal any difference between I and I' is by expanding n. Since it did not, it will not find solution path P^*, but will halt with cost $C^* > D$, the same cost it found for I and worse than that found by A*. This contradicts its property of being globally compatible with A*.

□

Corollary 4:
A* is 0-optimal over $\underline{A}_{gc}$ relative to $\underline{I}_{AD}^{-}$.

The corollary follows from the fact that in non-pathological instances A* expands *only* surely expanded nodes.

Corollary 5:
A* is 1-optimal over $\underline{A}_{gc}$ relative to $\underline{I}_{AD}$.

Proof:
The proof relies on the observation that for every optimal path P^* (in any instance of $\underline{I}_{AD}$) there is a tie-breaking-rule of A* that expands only nodes along P^* plus perhaps some other nodes having $g(n)+h(n)<C^*$, i.e., the only nodes expanded satisfying the equality $g(n)+h(n)=C^*$ are those on P^*. Now, if A* is not 1-optimal over A_{gc} then, given an instance I, there exists an algorithm $B \in A_{gc}$ such that B avoids some node expanded by all tie-breaking-rules in A*. To contradict this supposition let A_1^* be the tie-breaking-rule of A* that returns the same optimal path P^* as B returns, but expands no node outside P^* for which $g(n)+h(n)=C^*$. Clearly, any node n which B avoids and A_1^* expands must satisfy $g(n)+h(n)<C^*$. We can now apply the argument used in the proof of Theorem 4, appending to n a branch to a goal node t', with cost $c(n,t')=h(n)$. Clearly, A_1^* will find the optimal path (s,n,t') costing $g(n)+h(n)<C^*$, while B will find the old path costing C^*, thus violating its global compatibility with A*.

□

4.3 Optimality Over Best-First Algorithms, A_{bf}

The next result establishes A*'s optimality over the set of best-first algorithms which are admissible if provided with $h \le h^*$. These algorithms will be permitted to employ any evaluation function f_P where f is a function of the nodes, the edge-costs, and the heuristic function h evaluated on the nodes of P, i.e.

$$f_P \triangleq f(s,n_1,n_2,\ldots,n) = f(\{n_i\},\{c(n_i,n_{i+1})\},\{h(n_i)\} \mid n_i \in P). \quad (17)$$

Lemma 1:
Let B be an admissible Best-First algorithm using an evaluation function f_B such that for every $(G,s,\Gamma,h) \in \underline{I}_{AD}$ f_B satisfies:

$$f_{P_i^s}=f(s,n_1,n_2,\ldots,\gamma)=C(P_i^s)\quad \forall\gamma\in\Gamma. \tag{18}$$

Where P_i^s as any solution path in G. If B is admissible on $\underline{I}_{AD}$, then $\underline{N}_{g+h}^{C^*}\subseteq\underline{N}_{f_B}^{C^*}$. That is, the set of nodes reachable from s by a path whose $g+h$ values are strictly less then C^* is contained in those who are reachable by a path whose f_B values are strictly bounded by C^*.

Proof:

Let $I=(G,s,\Gamma,h)\in\underline{I}_{AD}$ and assume $n\in\underline{N}_{g+h}^{C^*}$ but $n\notin\underline{N}_{f_B}^{C^*}$, i.e., there exists a path P_{s-n} such that for every $n'\in P_{s-n}$
$g_P(n')+h(n')<C^*$ and for some $n'\in P_{s-n}$ $f_B(n')\geq C^*$.

Let

$$Q=\max_{n'\in P_{s-n}}\left\{g(n')+h(n')\right\} \tag{19}$$

$$Q_B=\max_{n'\in P_{s-n}}\left\{f_B(n')\right\} \tag{20}$$

Obviously: $Q<C^*$ and $Q_B\geq C^*\Rightarrow Q_B>Q$. Define G' to include path P_{s-n} with two additional goal nodes t_1,t_2 as described by figure 5. The cost on edge (s,t_2) is $\frac{Q_B+Q}{2}$, the cost on edge (n,t_1) is $Q-g_{P_{s-n}}(n)$, t_1 and t_2 are assigned $h'=0$ while all other nodes retain their old h. $I'=(G',s,\Gamma\cup\{t_1,t_2\},h)$ $\in\underline{I}_{AD}$ since $\forall n'$, $n'\in P_{s-n}$, $g(n')+h(n')\leq Q$ which implies that $h(n')\leq Q-g(n')=h^*_{I'}(n')$.

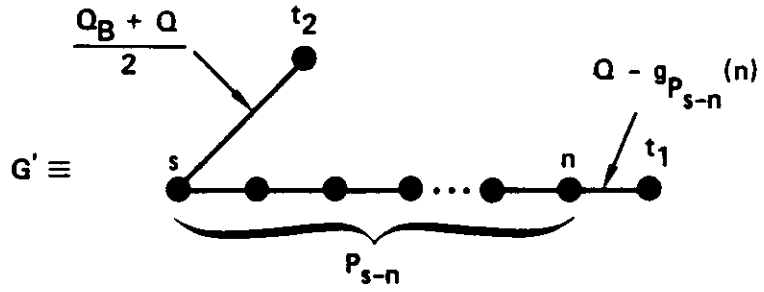

Figure 5 - A graph showing that any admissible best-first algorithm must assign $f_B(n)<C^*$ to every node n along P_{s-n}.

Obviously the optimal path in G' is P_{s-t_1} with cost Q. However, the evaluation function f_B satisfies

$$f_B(t_2) = C(P_{s-t_2}) = \frac{Q_B + Q}{2} < Q_B \qquad (21)$$

Therefore, B, as a Best-First algorithm, will expand t_2 before expanding n implying that B halts on the suboptimal path P_{s-t_2}, thus contradicting its admissibility.

□

Theorem 5:
Let B be a Best-First algorithm such that f_B satisfies the property of Lemma 1.

a. If B is admissible over $\underline{I}_{AD}$ then B expands every node in $\underline{N}^{C^*}_{g+h}$.

b. If B is admissible over $\underline{I}_{AD}$ and f_B is of the form:

$$f_{P_{s-n}}(n) = F(g_{P_{s-n}}(n), h(n)) \qquad (22)$$

then $F(x,y) \leq x+y$.

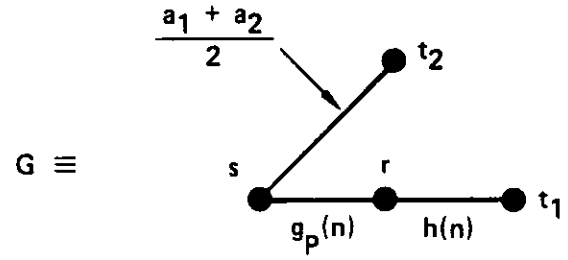

Figure 6 - A graph demonstrating that an admissible best-first algorithm cannot assign to any node a value $f(n) = F[g(n), h(n)]$ greater then $g(n) = h(n)$.

Proof:

a. A Best-First algorithm that satisfy the property of lemma 1 must expand all nodes in $N^{C^*}_{f_B}$. This is so because immediately before the algorithm terminates it expands a goal node whose f_B value is equal to the cost of the path (due to condition (18)) and this cost is no smaller then C^*. Therefore all paths strictly bounded by C^* must have been selected for expansion before. From that and from lemma 1 it follows that an admissible B expands all the nodes in $N^{C^*}_{g+h}$.

b. Assume to the contrary that there is a path P and a node $n \in P$ such that

$$F(g_P(n),h(n)) > g_P(n)+h(n) \qquad (23)$$

Let $a_1=F(g_P(n),h(n))$, $a_2=g_P(n)+h(n)$. Obviously, $a_2 < \frac{a_1+a_2}{2} < a_1$. Let G be a graph as shown in Figure 6 having nodes s,r,t_1,t_2 and edges (s,r), (r,t_1), (s,t_2) with costs $C(s,r)=g_P(n)$, $C(r,t_1)=h(n)$, $C(s,t_2)=\frac{a_1+a_2}{2}$. Let $I=(G,s,\{t_1,t_2\},h)$ where $h(s)=0$, $h(r)=h(n)$ and $h(t_1)=h(t_2)=0$. Obviously $I \in \underline{I}_{AD}$. However,

$$f_B(r) = F(g_P(n),h(n)) = a_1 > \frac{a_1+a_2}{2} = c(s,t_2) = f_B(t_2) \qquad (24)$$

implying that B halts on solution path P_{s-t_2}, again contradicting its admissibility.

□

Corollary 6:
A* is 0-optimal over $\underline{A}_{bf}$ relative to $\underline{I}_{\overline{AD}}$ and 1-optimal relative to $\underline{I}_{CON}$. A** is 1-optimal over $\underline{A}_{bf}$ relative to $\underline{I}_{AD}$.

□

An interesting implication of Part b of Theorem 5 asserts that any admissible combination of g and h, $h \le h^*$, will expand every node surely expanded by A*. In other words, the additive combination $g+h$ is, in this sense, the optimal way of aggregating g and h for additive cost measures.

The 0-optimality of A* relative to nonpathological instances of $\underline{I}_{AD}$ also implies that in these instances $g(n)$ constitutes a sufficient summary of the information gathered along the path from s to n. Any additional information regarding the heuristics assigned to the ancestors of n, or the costs of the individual arcs along the path, is superfluous, and cannot yield a further reduction in the number of nodes expanded. Such information, however, may help reduce the number of node *evaluations* performed by the search (see [Martelli 1977, Bagchi 1983] and [Mero 1984].).

4.4 Summary And Discussion

Our results concerning the optimality of A* are summarized in Table 1. For each class-domain combination from Figure 2, the table identifies the strongest type of optimality that exists and the algorithm achieving it.

Domain of Problem Instances	Class of Algorithms: Admissible if $h \leq h^*$ A_{ad}	Globally Compatible with A^* A_{gc}	Best-First A_{bf}
Admissible I_{AD}	A^{**} is 3-optimal No 2-optimal exists	A^* is 1-optimal No 0-optimal exists	A^* is 1-optimal No 0-optimal exists
Admissible and nonpathological I^-_{AD}	A^* is 2-optimal No 1-optimal exists	A^* is 0-optimal	A^* is 0-optimal
Consistent I_{CON}	A^* is 1-optimal No 0-optimal exists	A^* is 1-optimal No 0-optimal exists	A^* is 1-optimal No 0-optimal exists
Consistent nonpathological I^-_{CON}	A^* is 0-optimal	A^* is 0-optimal	A^* is 0-optimal

TABLE 1

The most significant results are those represented in the left-most column, relating to $\underline{A}_{ad}$, the entire class of algorithms which are admissible whenever provided with optimistic advice. Contrary to prevailing beliefs A* turns out not to be optimal over $\underline{A}_{ad}$ relative to every problem graph quantified with optimistic estimates. There are admissible algorithms which, in some graphs, will find the optimal solution in just a few steps whereas A* (as well as A** and all their variations) would be forced to explore arbitrary large regions of the search graphs (see Fig. 4a). In bold defiance of Hart, Nilsson, and Raphael's [Hart 1968] argument for A*'s optimality, these algorithms succeed in outsmarting A* by penetrating regions of the graph that A* finds unpromising (at least temporarily), visiting some goal nodes there, then processing the information gathered to identify and purge those nodes on OPEN which no longer promise to sprout a solution better than the cheapest one at hand.

In nonpathological cases, however, these algorithms cannot outsmart A* without paying a price. The 2-optimality of A* relative to $\underline{I}_{AD}$ means that each such algorithm must always expand at least one node which A* will skip. This means that the only regions of the graph capable of providing node-purging information are regions which A* will not visit *at all*. In other words, A* makes full use of the information gathered along its search and there could be no gain in changing the order of visiting nodes which A* plans to visit anyhow.

This instance-by-instance node tradeoff no longer holds when pathological cases are introduced. The fact that A* is not 2-optimal relative to $\underline{I}_{AD}$ means that some smart algorithms may outperform A* by simply penetrating certain regions of the graph *earlier than A** (A* will later visit these regions), thus expanding only a proper subset of the set of nodes expanded by A*. In fact the lack of 2-optimality in the $(\underline{A}_{ad}, \underline{I}_{AD})$ entry of table 1 means that no algorithm can be protected against such smart competitors; For any admissible algorithm A_1, there exists another admissible algorithm A_2 and a graph G quantified by optimistic heuristic h ($h \leq h^*$) such that A_2 expands fewer nodes than A_1 when applied to G. Mero [Mero 1984] has recently shown that no optimal algorithm exists if complexity is measured by the number of expansion operations (a node can be reopened several times). Our result now shows that $\underline{A}_{ad}$ remains devoid of an optimal algorithm even if we measure complexity by the number of distinct nodes expanded.

The type-3 optimality of A** over $\underline{A}_{ad}$ further demonstrates that those 'smart' algorithms which prevent A* from achieving optimality are not smart after all, but simply lucky; each takes advantage of the peculiarity of the graph for which it was contrived and none can maintain this superiority over all problem instances. If it wins on one graph there must be another where it is beaten, and by the very same opponent, A**. It is in this sense that A** is 3-optimal, it exhibits a universal robustness against all its challengers.

Perhaps the strongest claim that Table 1 makes in favor of A* is contained in the entries related to I_{CON}, the domain of problems in which h is known to be not only admissible, but also consistent. It is this domain that enables A* to unleash its full pruning powers, achieving a node-by-node superiority (types 0, if non-pathological and otherwise type 1) over all admissible algorithms. Recalling also that, under consistent h, A* never reopens closed nodes and that only few nodes are affected by the choice of tie-breaking-rule (see [Pearl 1984b]), we conclude that in this domain A* constitutes the most effective scheme of utilizing the advice provided by h.

This optimality is especially significant in light of the fact that consistency is not an exception but rather a common occurrence; almost all admissible heuristics invented by people are consistent. The reason is that the technique people invoke in generating heuristic estimates is often that of *relaxation*; we imagine a simplified version of the problem at hand by relaxing some of its constraints, solve the relaxed version mentally, then we use the cost of the resulting solution as a heuristic for the original problem [Pearl 1983] It can be shown that any heuristic generated by such a process is automatically consistent, which explains the abandon of consistent cases among human-generated heuristics. Thus, the strong optimality of A* under the guidance of consistent heuristics implies, in effect, its optimality in most cases of practical interest.

APPENDIX : Properties of A**

Algorithm A** is a variation of A*. It can be viewed as a Best-First algorithm that uses an evaluation function:

$$f'_{P_{s-n}}(n) = \max\left\{ f(n') = g_{P_{s-n}}(n') + h(n') \mid n' \in P_{s-n} \right\}$$

A** differs from A* in that it relies not only on the $g+h$ value of node n, but also considers the $g+h$ values along the path from s to n. The maximum is then used as a criterion for node selection. Note that A** cannot be considered a BF* algorithm since it uses one function, f', for ordering nodes for expansion (step 3) and a *different* function g for redirecting pointers (step 6c). Had we allowed A** to use f' for both purposes it would not be admissible relative to $\underline{I}_{AD}$, since f' is not order preserving.

We will now show that A** is admissible over $\underline{I}_{AD}$.

Theorem:
Algorithm A** will terminate with an optimal solution in every problem instance where $h \leq h^*$

Proof:
Suppose the contrary. Let C be the value of the path P_{s-t} found by A** and assume $C > C^*$.

We will argue that exactly before A** chooses the goal node t for expansion, there is an OPEN node n' on an optimal path with $f'(n') \leq C^*$. If we show that, than obviously A** should have selected n' for expansion and not t, since $f'(n') \leq C^* < C = f'(t)$, which yields a contradiction.

Since A** redirects pointers according to g, the pointer assigned to the shallowest OPEN node n' along any optimal path is directed along that optimal path. Moreover, $h \leq h^*$ implies that such a node satisfies $f'(n') \leq C^*$, and this completes our argument. □

a) For every tie-breaking-rule of A* and for every problem instance $I \in \underline{I}_{AD}$, there exists a tie-breaking-rule for A** which expands a subset of the nodes expanded by A*. Moreover,

b) There exists a problem instance and a tie-breaking-rule for A** that expands a proper subset of the nodes which are expanded by any tie-breaking-rule of A*.

Proof: Part a

From the definition of f' it is clear that all paths which are strictly bounded below C^* relative to f are also strictly bounded below C^* relative to f'. Therefore, both algorithms have exactly the same set of surely expanded nodes, $\underline{N}_f^{C^*} = \underline{N}_{f'}^{C^*}$, and this set is expanded before any node outside this set. Let n^* be the first node expanded by A* satisfying the equality $f(n^*)=C^*$. Exactly before n^* is chosen for expansion all nodes in $\underline{N}_f^{C^*}$ were already expanded. A**, after expanding those nodes also has n^* in OPEN with $f'(n^*)=C^*$; there exists, therefore, a tie-breaking-rule in A** which will also choose n^* for expansion. From that moment on, A* will expand some sequence $n^*, n_1, n_2, n_3, \cdots t$ for which $f(n_i) \leq C^*$. Since on these nodes $f'(n_i) = C^*$, it is possible for A** to use a tie-breaking-rule that expands exactly that same sequence of nodes until termination.

Part b

Examine the graph of Figure 7.

n_1 and n_2 will be expanded by every tie-breaking-rule of A* while there is a tie-breaking-rule for A** that expands only P_{s-t}.

□

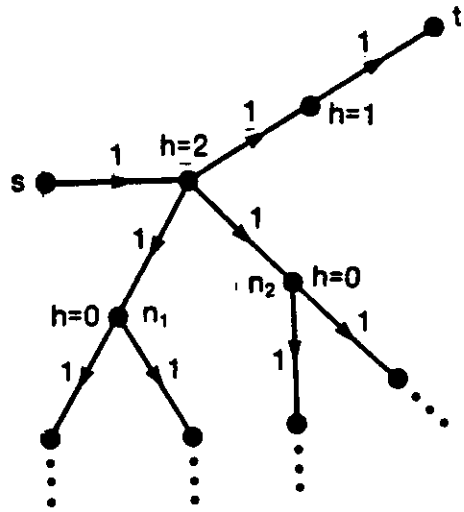

Figure 7

References

[Bagchi 1983] A. Bagchi and A. Mahanti, "Search algorithms under different kinds of heuristics - A comparative study.," *Journal of the ACM*, Vol. 30, No. 1, 1983, pp. 1-21.

[Barr 1981] A. Barr and E.A. Feigenbaum, *Handbook of Artificial Intelligence*, Los Altos, California: William Kaufman Inc., 1981.

[Dechter 1985] R. Dechter and J. Pearl, "Generalized best-first search strategies and the optimality of A*.," *Journal of the ACM,*, Vol. 32, No. 3, 1985, pp. 505-536.

[Gelperin 1977] D. Gelperin, "On the optimality of A*," *Artificial Intelligence*, Vol. 8, No. 1, 1977, pp. 69-76.

[Harris 1974] L.R. Harris, "The heuristic search under conditions of error," *Artificial Intelligence*, Vol. 5, No. 3, 1974, pp. 217-234.

[Hart 1968] P.E. Hart, N.J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEE Trans. System Science and Cybernatics*, Vol. SSC-4, No. 2, 1968, pp. 100-107.

[Martelli 1977] A. Martelli, "On the Complexity of admissible search algorithms," *Artificial Intelligence*, Vol. 8, No. 1, 1977, pp. 1-13.

[Mero 1984] L. Mero, "A heuristic search algorithm with modifiable estimate," *Artificial Intelligence*, Vol. 23, No. 1, 1984, pp. 13-27.

[Nilsson 1980] N. Nilsson, *Principals of Artificial Intelligence*, Palo Alto, California: Tioga, 1980.

[Nilsson 1971.] N.J. Nilsson, *Problem-Solving Methods in Artificial Intelligence*: New York , McGraw-Hill., 1971..

[Pearl 1983] J. Pearl, "On the discovery and generation of certain heuristics," *AI Magazine*, No. 22-23, 1983.

[Pearl 1984a] J. Pearl, "Some recent results in heuristic search theory," *IEEE Transaction on pattern analysis and machine intelligence*, Vol. PAMI-6, No. 1, 1984, pp. 1-13.

[Pearl 1984b] J. Pearl, *HEURISTICS: Intelligent Search Strategies for Computer Problem Solving*, reading Mass: Addison-Wesley, 1984.

[Pohl 1971] I. Pohl, "Bi-directional Search," in *Machine Intelligence 6*, B. Meltzer D. Michie, Ed. New York: American Elsevier: 1971, pp. 127-140.