It is beyond the scope of the discussion to consider how to define a successor operator Γ or assign costs c_{ij} so that the resulting graph realistically reflects the nature of a specific problem domain.[2]

B. The Heuristic Power of the Estimate $\hat{h}$

The algorithm A^* is actually a family of algorithms; the choice of a particular function $\hat{h}$ selects a particular algorithm from the family. The function $\hat{h}$ can be used to tailor A^* for particular applications.

As was discussed above, the choice $\hat{h} = 0$ corresponds to the case of knowing, or at least of using, absolutely no information from the problem domain. In our example of cities connected by roads, this would correspond to assuming a priori that roads could travel through "hyperspace," i.e., that any city may be an arbitrarily small road distance from any other city regardless of their geographic coordinates.

Since we are, in fact, "more informed" about the nature of Euclidean space, we might increase $\hat{h}(n)$ from 0 to $\sqrt{x^2 + y^2}$ (where x and y are the magnitudes of the differences in the x, y coordinates of the city represented by node n and its closest goal city). The algorithm would then still find the shortest path, but would do so by expanding, typically, considerably fewer nodes. In fact, A^* expands no more nodes[3] than any admissible algorithm that uses no more information from the problem domain; viz., the information that a road between two cities might be as short as the airline distance between them.

Of course, the discussion thus far has not considered the cost of computing $\hat{h}$ each time a node is generated on the graph. It could be that the computational effort required to compute $\sqrt{x^2 + y^2}$ is significant when compared to the effort involved in expanding a few extra nodes; the optimal procedure in the sense of minimum number of nodes expanded might not be optimal in the sense of minimum total resources expended. In this case one might, for example, choose $\hat{h}(n) = (x + y)/2$. Since $(x + y)/2 < \sqrt{x^2 + y^2}$, the algorithm is still admissible. Since we are not using "all" our knowledge of the problem domain, a few extra nodes may be expanded, but total computational effort may be reduced; again, each "extra" node must also be expanded by other admissible algorithms that limit themselves to the "knowledge" that the distance between two cities may be as small as $(x + y)/2$.

Now suppose we would like to reduce our computational effort still further, and would be satisfied with a solution path whose cost is not necessarily minimal. Then we could choose an $\hat{h}$ somewhat larger than the one defined by (3). The algorithm would no longer be admissible, but it might be more desirable, from a heuristic point of view, than any admissible algorithm. In our roads-and-cities example, we might let $\hat{h} = x + y$. Since road distance is usually substantially greater than airline distance, this $\hat{h}$ will usually, but not always, result in an optimal solution path. Often, but not always, fewer nodes will be expanded and less arithmetic effort required than if we used $\hat{h}(n) = \sqrt{x^2 + y^2}$.

Thus we see that the formulation presented uses one function, $\hat{h}$, to embody in a formal theory all knowledge available from the problem domain. The selection of $\hat{h}$, therefore, permits one to choose a desirable compromise between admissibility, heuristic effectiveness, and computational efficiency.

[2] We believe that appropriate choices for Γ and c_{ij} will permit many of the problem domains in the heuristic programming literature[7] to be mapped into graphs of the type treated in this paper. This could lead to a clearer understanding of the effects of "heuristics" that use information from the problem domain.

[3] Except for possible critical ties, as discussed in Corollary 2 of Theorem 3.

REFERENCES

[1] M. Pollack and W. Wiebenson, "Solutions of the shortest-route problem—a review," *Operations Res.*, vol. 8, March–April 1960.

[2] R. Busacker and T. Saaty, *Finite Graphs and Networks: An Introduction with Applications.* New York: McGraw-Hill, 1965, ch. 3.

[3] R. Bellman and S. Dreyfus, *Applied Dynamic Programming.* Princeton, N. J.: Princeton University Press, 1962.

[4] H. Gelernter, "Realization of a geometry-theorem proving machine," in *Computers and Thought.* New York: McGraw-Hill, 1963.

[5] A. Samuel, "Some studies in machine learning using the game of checkers," in *Computers and Thought.* New York: McGraw-Hill, 1963.

[6] E. Moore, "The shortest path through a maze," *Proc. Internat'l Symp. on Theory of Switching* (April 2–5, 1957), pt. 2. Also, *The Annals of the Computation Laboratory of Harvard University*, vol. 30. Cambridge, Mass.: Harvard University Press, 1959.

[7] E. Feigenbaum and J. Feldman, *Computers and Thought.* New York: McGraw-Hill, 1963.

The next theorem extends Theorem 2 to situations where ties may occur. It states that for any admissible algorithm A, one can always find a member A^* of $\mathcal{A}^*$ such that each node expanded by A^* is also expanded by A.

Theorem 3

Let A be any admissible algorithm no more informed than the algorithms in $\mathcal{A}^*$, and suppose the consistency assumption is satisfied by the $\hat{h}$ used in the algorithms in $\mathcal{A}^*$. Then for any δ graph G_s there exists an $A^* \epsilon\, \mathcal{A}^*$ such that every node expanded by A^* is also expanded by A.

Proof: Let G_s be any δ graph and A_1^* be any algorithm in $\mathcal{A}^*$. If every node of G_s that A_1^* expands is also expanded by A, let A_1^* be the A^* of the theorem. Otherwise, we will show how to construct the A^* of the theorem by changing the tie-breaking rule of A_1^*. Let L be the set of nodes expanded by A, and let $P = (s, n_1, n_2, \cdots, n_k, t)$ be the optimal path found by A.

Expand nodes as prescribed by A_1^* as long as all nodes selected for expansion are elements of L. Let n be the first node selected for expansion by A_1^* which is not in L. Now $\hat{f}(n) \leq f(s)$ by the corollary to Lemma 3. Since $\hat{f}(n) < f(s) = f(t)$ would imply that A is inadmissible (by the argument of Theorem 2), we may conclude that $\hat{f}(n) = f(s)$. At the time A_1^* selected n, goal node t was not closed (or A_1^* would have been terminated). Then by the corollary to Lemma 1, there is an open node n' on P such that $\hat{f}(n') \leq f(s) = \hat{f}(n)$. But since n was selected for expansion by A_1^* instead of n', $\hat{f}(n) \leq \hat{f}(n')$. Hence $\hat{f}(n) \leq \hat{f}(n') \leq \hat{f}(n)$, so $\hat{f}(n) = \hat{f}(n')$. Let A_2^* be identical to A_1^* except that the tie-breaking rule is modified just enough to choose n' instead of n. By repeating the above argument, we obtain for some i an $A_i^* \epsilon\, \mathcal{A}^*$ that expands only nodes that are also expanded by A, completing the proof of the theorem.

Corollary 1

Suppose the premises of the theorem are satisfied. Then for any δ graph G_s there exists an $A^* \epsilon\, \mathcal{A}^*$ such that $N(A^*, G_s) \leq N(A, G_s)$, with equality if and only if A expands the identical set of nodes as A^*.

Since we cannot select the most fortuitous tie-breaking rule ahead of time for each graph, it is of interest to ask how all members of $\mathcal{A}^*$ compare against any admissible algorithm A in the number of nodes expanded. Let us define a *critical* tie between n and n' as one for which $\hat{f}(n) = \hat{f}(n') = f(s)$. Then we have the following as a second corollary to Theorem 3.

Corollary 2

Suppose the premises of the theorem are satisfied. Let $R(A^*, G_s)$ be the number of critical ties which occurred in the course of applying A^* to G_s. Then for any δ graph G_s and *any* $A^* \epsilon \mathcal{A}^*$,

$$N(A^*, G_s) \leq N(A, G_s) + R(A^*, G_s).$$

Proof: For any noncritical tie, all alternative nodes must be expanded by A as well as by A^* or A would not be admissible. Therefore, we need merely observe that each node expanded by A^* but not by A must correspond to a different critical tie in which A^*'s tie-breaking rule made the inappropriate choice.

Of course, one must remember that when A does expand fewer nodes than some particular A^* in $\mathcal{A}^*$, it is only because A was in some sense "lucky" for the graph being searched, and that there exists a graph consistent with the information available to A and A^* for which A^* would not search more nodes than A.

Note that, although one cannot keep a running estimate of R while the algorithm proceeds because one does not know the value of $f(s)$, this value is established as soon as the algorithm terminates, and R can then be easily computed. In most practical situations, R is not likely to be large because critical ties are likely to occur only very close to termination of the algorithm, when $\hat{h}$ can become a perfect estimator of h.

IV. Discussion and Conclusions

A. Comparisons Between A^* and Other Search Techniques

Earlier we mentioned that the estimate $\hat{h}(n) \equiv 0$ for all n trivially satisfies the consistency assumption. In this case, $\hat{f}(n) = \hat{g}(n)$, the lowest cost so far discovered to node n. Such an estimate is appropriate when no information at all is available from the problem domain. In this case, an admissible algorithm cannot rule out the possibility that the goal might be as close as δ to that node with minimum $g(n)$. Pollack and Wiebenson[1] discuss an algorithm, proposed to them by Minty in a private communication, that is essentially identical to our A^* using $\hat{f}(n) = \hat{g}(n)$.

Many algorithms, such as Moore's "Algorithm D"[6] and Busacker and Saaty's implementation of dynamic programming, keep track of $\hat{g}(n)$ but do not use it to order the expansion of nodes. The nodes are expanded in a "breadth-first" order, meaning that all nodes one step away from the start are expanded first, then all nodes two steps away, etc. Such methods must allow for changes in the value of $\hat{g}(n)$ as a node previously expanded is later reached again by a less costly route.

It might be argued that the algorithms of Moore, Busacker and Saaty, and other equivalent algorithms (sometimes known as "water flow" or "amoeba" algorithms) are advantageous because they first encounter the goal by a path with a minimum number of steps. This argument merely reflects an imprecise formulation of the problem, since it implies that the number of steps, and not the cost of each step, is the quantity to be minimized. Indeed, if we set $c_{ij} = 1$ for all arcs, this class of algorithms is identical to A^* with $\hat{h} = 0$. We emphasize that, as is always the case when a mathematical model is used to represent a real problem, the first responsibility of the investigator is to ensure that the model is an adequate representation of the problem for his purposes.

$$g(n) = g(n') + h(n',n)$$
$$= \hat{g}(n') + h(n',n).$$

Thus,

$$\hat{g}(n) > \hat{g}(n') + h(n',n).$$

Adding $\hat{h}(n)$ to both sides yields

$$\hat{g}(n) + \hat{h}(n) > \hat{g}(n') + h(n',n) + \hat{h}(n).$$

We can apply (5) to the right-hand side of the above inequality to yield

$$\hat{g}(n) + \hat{h}(n) > \hat{g}(n') + \hat{h}(n')$$

or

$$\hat{f}(n) > \hat{f}(n'),$$

contradicting the fact that A^* selected n for expansion when n' was available and thus proving the lemma.

The next lemma states that $\hat{f}$ is monotonically nondecreasing on the sequence of nodes closed by A^*.

Lemma 3

Let $(n_1, n_2, \cdots, n_t)$ be the sequence of nodes closed by A^*. Then, if the consistency assumption is satisfied, $p \leq q$ implies $\hat{f}(n_p) \leq \hat{f}(n_q)$.

Proof: Let n be the next node closed by A^* after closing m. Suppose first that the optimum path to n does not go through m. Then n was available at the time m was selected, and the lemma is trivially true. Then suppose that the optimum path to n does, in fact, go through m. Then $g(n) = g(m) + h(m, n)$. Since, by Lemma 2, we have $\hat{g}(n) = g(n)$ and $\hat{g}(m) = g(m)$,

$$\hat{f}(n) = \hat{g}(n) + \hat{h}(n)$$
$$= g(n) + \hat{h}(n)$$
$$= g(m) + h(m, n) + \hat{h}(n)$$
$$\geq g(m) + \hat{h}(m)$$
$$= \hat{g}(m) + \hat{h}(m)$$

where the inequality follows by application of (5). Thus we have

$$\hat{f}(n) \geq \hat{f}(m).$$

Since this fact is true for any pair of nodes n_k and n_{k+1} in the sequence, the proof is complete.

Corollary

Under the premises of the lemma, if n is closed then $\hat{f}(n) \leq f(s)$.

Proof: Let t be the goal node found by A^*. Then $\hat{f}(n) \leq \hat{f}(t) = f(t) = f(s)$.

We can now prove a theorem about the optimality of A^* as compared with any other admissible algorithm A that uses no more information about the problem than does A^*. Let $\Theta_n{}^A$ be the index set used by algorithm A at node n. Then, if $\Theta_n{}^{A^*} \subset \Theta_n{}^A$ for all nodes n in G_s, we shall say that algorithm A is *no more informed* than algorithm A^*.

The next theorem states that if an admissible algorithm A is no more informed than A^*, then any node expanded by A^* must also be expanded by A. We prove this theorem for the special case for which ties never occur in the value of $\hat{f}$ used by A^*. Later we shall generalize the theorem to cover the case where ties can occur, but the proof of the no-ties theorem is so transparent that we include it for clarity.

Theorem 2

Let A be any admissible algorithm no more informed than A^*. Let G_s be any δ graph such that $n \neq m$ implies $\hat{f}(n) \neq \hat{f}(m)$, and let the consistency assumption be satisfied by the $\hat{h}$ used in A^*. Then if node n was expanded by A^*, it was also expanded by A.

Proof: Suppose the contrary. Then there exists some node n expanded by A^* but not by A. Let t^* and t be the preferred goal nodes of s found by A^* and A, respectively. Since A^* and A are both admissible,

$$\hat{f}(t^*) = \hat{g}(t^*) + \hat{h}(t^*) = g(t^*) + 0 = f(t^*) = f(t) = f(s).$$

Since A^* must have expanded n before closing t^*, by Lemma 3 we have

$$\hat{f}(n) < \hat{f}(t^*) = f(t).$$

(Strict inequality occurs because no ties are allowed.)

There exists some graph $G_{n,\theta}$, $\theta \in \Theta_n$, for which $\hat{h}(n) = h(n)$ by the definition of $\hat{h}$. Now by Lemma 2, $\hat{g}(n) = g(n)$. Then on the graph $G_{n,\theta}$, $\hat{f}(n) = f(n)$. Since A is no more informed than A^*, A could not rule out the existence of $G_{n,\theta}$; but A did not expand n before termination and is, therefore, not admissible, contrary to our assumption and completing the proof.

Upon defining $N(A,G_s)$ to be the total number of nodes in G_s expanded by the algorithm A, the following simple corollary is immediate.

Corollary

Under the premises of Theorem 2,

$$N(A^*, G_s) \leq N(A, G_s)$$

with equality if and only if A expands the identical set of nodes as A^*.

In this sense, we claim that A^* is an optimal algorithm. Compared with other no more informed admissible algorithms, it expands the fewest possible nodes necessary to guarantee finding an optimal path.

In case of ties, that is if there exist two or more open nodes $n_1, \cdots, n_k$ with $\hat{f}(n_1) = \cdots = \hat{f}(n_k) < \hat{f}(n)$ for every other open node n, A^* arbitrarily chooses one of the n_i. Consider the set $\mathcal{A}^*$ of all algorithms that act identically to A^* if there are no ties, but whose members resolve ties differently. An algorithm is a member of $\mathcal{A}^*$ if it is simply the original A^* with any arbitrary tie-breaking rule.