A New Admissible Heuristic for Minimal-Cost Proofs*

Eugene Charniak and **Saadia Husain**
Department of Computer Science Brown University
Box 1910, Providence RI 02912

Abstract

Finding best explanations is often formalized in AI in terms of minimal-cost proofs. Finding such proofs is naturally characterized as a best-first search of the proof-tree (actually a proof dag). Unfortunately the only known search heuristic for this task is quite poor. In this paper we present a new heuristic, a proof that it is admissible (for certain successor functions), and some experimental results suggesting that it is a significant improvement over the currently used heuristic.

Introduction

The problem of explanation (or abduction) is often formalized within AI as a problem of proving the things to be explained using some auxiliary hypotheses [1, 5,7,9,10,11]. For example, a circuit's showing a particular fault is explained by proving that the circuit would show the fault if a particular component, say Component-5, were broken. We have thus explained the fault via the auxiliary hypotheses, "Component-5 is broken."

A persistent problem with such approaches is that there will typically be many such proofs based upon different sets of auxiliary hypotheses. One can use the criterion of set minimality to remove some of these, but still the numbers remain large. This naturally suggests that we should somehow weight the proofs and look for a proof which is optimal in some sense. If we talk about the "costs" of proofs, and then pick the one with the smallest cost, we are lead to the problem of finding the "minimal cost proofs" of our title.

A clean axiomatization of this idea, called "cost-based abduction," is presented in [4]. Auxiliary hypotheses have associated costs, and the cost of a proof is the sum of the costs of the extra hypotheses required to make the proof go through. This is quite close to what is done by Hobbs and Stickel [7]. It is shown in [4,12] how the costs can be given a firm semantics in terms of probability theory, and that, as one might expect, finding minimal-cost proofs is NP-hard.

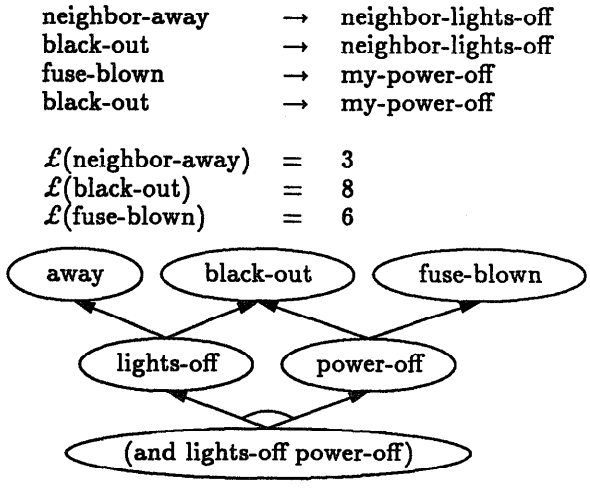

Figure 1: Some simple rules, and their and-or dag

As an example of such proofs, consider Figure 1. We show there some simple rules, the costs of various hypotheses (using the cost-function £), and the and-or dag showing the possible proofs of the observations "neighbors-lights-off" and "my-power-off". The minimal cost proof would assume "black-out" ($£ = 8$) despite the fact that it is the most expensive hypotheses. This occurs because the two facts which we need to explain can be thought of as "sharing" the cost of the hypothesis.

The use of an and-or dag to illustrate possible proofs suggest that one might find minimal-cost proofs by doing a best-first search of the and-or dag. How this could be done is outlined in [4]. The basic idea is that one starts with the expression to be proven, and then creates alternative partial proofs. In each iteration of the search algorithm a partial proof is chosen to expand, and it is expanded by considering how one of its goals can be satisfied. Finally one of the partial proofs

*Thanks to Philip Klein for helpful discussions,Solomon Shimony and Robert Goldman for reading an earlier draft of this paper, and to Eugene Santos for curve fitting. This work has been supported by the National Science Foundation under grant IRI-8911122 and by the Office of Naval Research, under contract N00014-88-K-0589.

will be completed by having all of its goals satisfied by either known facts, or auxiliary hypotheses. (We consider known facts to be zero-cost auxiliary hypotheses.)

Naturally, the efficiency of such a search will depend on having a good heuristic function for deciding which partial proof to work on next. Furthermore, we are only guaranteed that the first proof is the minimal-cost proof if the heuristic is admissible. Unfortunately, the only known admissible heuristic (indeed, the only known heuristic of any sort) for minimal-cost proofs is "cost so far." That is, given a partial proof, we estimate the cost of the total proof to be the actual costs incurred by the partial proof. This is the heuristic used in [4] and [13].

"Cost-so-far" is admissible, but otherwise it is a very bad heuristic. All of the costs in a minimal-cost proof are associated with the auxiliary hypotheses, which are, of course, at the leaves of the and-or dag. Thus it is typically only when the partial-proof is almost complete that the heuristic gives good guidance.

In this paper we present a better heuristic for minimal-cost proofs, and we prove that it is admissible if the partial-proofs are expanded in a certain way. We also give some preliminary results on how well the new heuristic behaves compared to "cost-so-far." Section 2 introduces the heuristic. We also show that it is not completely obvious that the heuristic is, in fact, admissible. (Indeed, we show that if not used properly it is inadmissible.) Section 3 proves the heuristic admissible when used with a certain class of successor functions. Section 4 gives the experimental results.

The New Heuristic

We first introduce our notation.

Definition 1 An *and-or dag* δ is a connected directed acyclic graph with nodes N_δ ($n \in N_\delta$), edges E_δ ($e \in E_\delta$), leaves $L_\delta (L_\delta \subset N_\delta, l \in L_\delta)$, and a single root $r_\delta \in N_\delta$. In general, capital letters denote sets and lower case denote individuals. We use "E" for edges, "L" for leaves, and "N" for nodes. To indicate parent and child relations in the dag we use subscript and superscript along with the convention that parents are "below" children. For example, X_y would be the set of x's immediately above y (i.e., y's children) whereas y^X would be the y immediate below all of the x's. As we saw in Figure 1, L_δ are the auxiliary hypotheses which we introduce to complete the proof, and r_δ is the theorem to be proven. We occasionally need sequence of edge sets. We use F_i for the i'th member of the sequence as E_i could be interpreted as the set of edges above node i.

Definition 2 Let δ be an and-or dag. An *and-dag* (from δ) is a dag obtained from δ by removing all but one of the descendants from every or-node in δ. We denote the set of all such dags by $\mathcal{A}(\delta)$. Intuitively the and-dags of δ correspond to the possible complete proofs for r_δ. For example Figure 2 shows two and-dags for the dag in Figure 1.

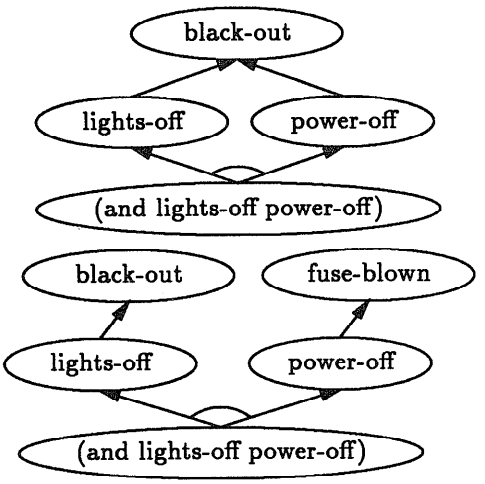

Figure 2: Two and-dags

Definition 3 A *weighted and-or dag* (Waodag) is a pair $(\delta, \pounds)$, with an and-or dag δ, and a *cost function* $\pounds : L_\delta \rightarrow \Re$ (the non-negative reals). In what follows all and-or dags are Waodags, and we leave the cost function implicit, thus referring to the Waodag by the term denoting the dag itself, e.g., δ.

Since an and-dag corresponds to a proof, the cost of an and-dag is naturally defined as the sum of the cost of the leaves. Since we want a minimal-cost proof for the entire dag we define a dag's cost to be the minimum cost of all of its and-dags.

Definition 4 Let δ be a Waodag and $\alpha \in \mathcal{A}(\delta)$. The leaves of α are L_α. We define the cost of δ as follows:

$$\begin{aligned} \pounds\alpha &= \sum_{l \in L_\alpha} \pounds l \\ \pounds\delta &= \min_{\alpha \in \mathcal{A}(\delta)} \pounds\alpha. \end{aligned}$$

Now we define our heuristic cost estimation function \$. The basic idea is that we want to get some estimate at lower nodes of the cost of the leaves above them. Thus the estimator will use values passed down from the leaves. Since, as we saw in Figure 1, the cost of a node can be "shared," we divide the cost of a node by how many parents it has, and that is the cost passed down to the parents.

Definition 5 Let δ be a Waodag. The *estimator* $\$: N_\delta \cup E_\delta \cup 2^{E_\delta} \rightarrow \Re$ is defined as follows:

$$\begin{aligned} \$n &= \begin{cases} \pounds n & n \in L_\delta \\ \$E_n & n \text{ is an and-node} \\ \min_{e \in E_n} \$e & n \text{ is an or-node} \end{cases} \\ \$e^n &= \frac{\$n}{|E^n|} \\ \$E &= \sum_{e \in E} \$e. \end{aligned}$$

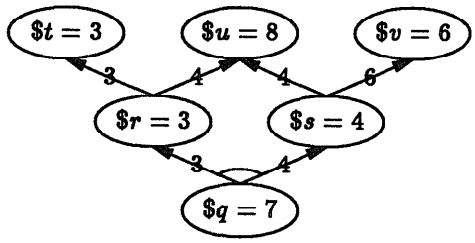

Figure 3: Estimated costs

Order of Derivation	Goals	Leaves	Estm Cost	Derived From
1	$\{q\}$	$\{\}$	7	
2	$\{rs\}$	$\{\}$	7	1
3	$\{s\}$	$\{t\}$	7	2
4	$\{s\}$	$\{u\}$	12	2
5	$\{\}$	$\{tv\}$	9	3

Figure 4: Five expansions of a Waodag

(We define the estimate for sets of edges because we will shortly be defining our expansion function in terms of sets of edges, so the estimate should be over these sets. See Figure 5.) Figure 3 shows the estimated costs for the nodes and edges of Figure 1 (where we have replaced the proposition symbols by individual letters to shorten things).

While it may seem reasonable that \$ is an admissible heuristic cost estimate, it is not necessarily so. Figure 4 shows five initial expansions in a best-first search for the minimal-cost proof based upon the estimates in Figure 3. The "goals" are the nodes which have yet to be expanded, and the "leaves" are the leaves which have been reached. Note that the estimated cost for the partial solution number 4 is, in fact, higher than the final solution would be if we had pursued this partial solution. Because of this incorrect estimate the best-first search tries partial solution 3 next, and finishes it off for a total cost of 9, in partial solution 5. As we have already noted, the correct minimal cost proof here has a cost of 8.

The admissible version of our heuristic requires a few small changes from the version shown in Figure 4. First, rather than define partial solutions in terms of nodes, it turns out to be easier to define them in terms of edges. Secondly, we require that nodes be expanded in a topological order such that a node is not expanded if a predecessor in the order has not been expanded. Expansion consists of removing all the edges immediately below the expansion node, and adding in the edges above it. (The next section will give a more formal characterization of the algorithm.)

We have also taken the liberty of adding some "pseudoedges", one just below the root and one above each leaf. These are convenient to define the starting point

Order of Derivation	Goals	Leaves	Estm Cost	Derived From
1	$\{e^q\}$	$\{\}$	7	
2	$\{e^r_q e^s_q\}$	$\{\}$	7	1
3	$\{e^s_q e^t_r\}$	$\{\}$	7	2
4	$\{e^s_q e^u_r\}$	$\{\}$	8	2
5	$\{e^t_r e^u_s\}$	$\{\}$	7	3
6	$\{e^t_r e^v_s\}$	$\{\}$	9	3
7	$\{e^u_s\}$	$\{e_t\}$	7	5
8	$\{\}$	$\{e_t e_u\}$	11	7
9	$\{e^u_r e^u_s\}$	$\{\}$	8	4
10	$\{e^u_r e^v_s\}$	$\{\}$	10	4
11	$\{\}$	$\{e_u\}$	8	9

Figure 5: The correct solution to our problem

of the search (the pseudoedge just below the root) and the ending point (where we have only edges above the leaves). In Figure 5 we show the search for the best proof for our continuing example, but now according to our new scheme.

The Proof

First, we formally describe the "pseudoedges" mentioned at the end of the last section. We modify our δ's by adding an extra layer of nodes L'_α above the leaves. Each l' is connected to the corresponding l by a single edge e_l. We also add an extra node r'_δ which has as its single child r_δ. We define

$$\pounds l' = \$l' = \$e_l = \pounds l$$

It should be clear that the costs of the original dag δ and the estimates on all of its nodes are uneffected by this change, so we henceforth just talk about δ as if it had these extra nodes and edges all the time.

Definition 6 A cut of an and-dag α is a set of edges C of α such that

- it is not possible to go from the root of α to a leaf of α without going through one of the edges
- if any edge is removed from C it will no longer be a cut.

If δ is a general Waodag (not necessarily an and-dag) then C is a cut of δ iff C is a cut of some $\alpha \in \mathcal{A}(\delta)$.

Definition 7 Let α be an and-tree, and τ be a topological sort of N_α, $\tau = \{n_0 \ldots n_k\}$, such that no node comes before any of its parents. We will *not* include in τ any of the extra nodes we added on at the beginning of thus section. Thus $n_0 = r_\alpha$, not r'_α. We define a function $\mathcal{E}_\tau : 2^{E_\alpha} \rightarrow 2^{E_\alpha}$ as follows:

$$\mathcal{E}_\tau(E) = \{E - E^n\} \cup E_n$$

where n is the first element of τ such that some $e^n \in E$ and E_n is non-empty. $\mathcal{E}_\tau$ is undefined if no such n exists. Intuitively, $\mathcal{E}_\tau$ is a search successor function, but just defined for and-dags. For reasons which are made

clear by the next theorem, we call such $\mathcal{E}_\tau$'s "topological cut extenders." $\mathcal{E}_\tau^*$ denotes the reflexive transitive closure of $\mathcal{E}_\tau$. $\mathcal{E}_\tau^i(E)$ denotes the sequence of edge sets generated (in order) by successive applications of $\mathcal{E}_\tau$.

Definition 8 We say that the topological cut extender $\mathcal{E}_\tau$ when applied to the set of edges E "expands" node n iff for all $e \in E_n, e \notin E$ and $e \in \mathcal{E}_\tau(E)$.

Theorem 1 *Let α be an and-tree and $\mathcal{E}_\tau^i(\{e^{r_\alpha}\}) = \{F_0, \ldots, F_k\}$. The following are true:*

1. $F_0 = \{e^{r_\alpha}\}$
2. $F_i, 0 \leq i \leq k$ *is a cut.*
3. $E^{n_i} \subset F_i$
4. $\mathcal{E}_\tau$ *when applied to* F_i *expands node* n_i *of* τ.
5. $F_k = E_{L_\alpha}$.

The proof has been omitted because of space limitations. See [3].

Definition 9 We say that C is a "proper cut" of the and-dag α iff there is a topological ordering of α, τ, such that $C \in \mathcal{E}_\tau^*(\{e^{r_\alpha}\})$. C is a proper cut of an and-or dag δ iff it is a proper cut of an and-dag $\alpha \in \mathcal{A}(\delta)$.

Intuitively proper cuts of and-trees are the boundaries of partial proofs. The cut extension function takes us from partial proof to partial proof until we reach the leaves. At this point we have only defined the process for and-dags, since this is all we need for the following few theorems. Eventually we will define a similar function, our successor function for the best-first search, which is defined for and-or dags.

Theorem 2 *If α is an and-dag, then for any proper cut C of α, $\$C = £\alpha$*

Proof. Let τ be the ordering of α such that $C \in \mathcal{E}_\tau^*(\{r_\alpha\})$. By Theorem 1 $\mathcal{E}_\tau^i(\{e^{r_\alpha}\})$ is the sequence of proper cuts $\{\{e^{r_\alpha}\}, \ldots, E_{L_\alpha}\}$. We prove the theorem by induction on this sequence, starting with E_{L_α}.
Basis step. Consider $C = E_{L_\alpha} = \{e_l \mid l \in L_\alpha\}$:

$$\$C = \sum_{l \in L_\alpha} \$e_l = \sum_{l \in L_\alpha} £l = £\alpha$$

Induction step. We prove the theorem for C such that for all cuts C' after it in the sequence $\{\{e^{r_\alpha}\}, \ldots, E_{L_\alpha}\}$, $\$C' = £\alpha$.

Let C' be the cut immediately after C in the sequence. By the definition of $\mathcal{E}_\tau$, $C' = \{C - E^n\} \cup E_n$, where n is the earliest node in τ such that $e^n \in C$. Next we observe that

$$\$C' = £\alpha = \$C - \$E^n + \$E_n$$

The first equality follows from the induction hypothesis. The second from the definition of C' and the fact that for all of the edges $e^n \in E^n$, it is the case that $e^n \in C$. (We know this from Theorem 1, claim 3.) Now, rearranging the above equation we get

$$\$C = £\alpha + (\$E^n - \$E_n).$$

It remains to show that the last term is zero. First

$$\$E^n = \sum_{e^n \in E^n} \$e^n = \sum_{e^n \in E^n} \frac{\$n}{\mid E^n \mid} = \$n.$$

Second, consider n. It must either be an and-node or an or-node. Suppose it is an or-node. Then since there is only one edge above it (this is an and-tree)

$$\$n = \min_{e_n \in E_n} \$e_n = \$E_n$$

If n is an and-node, then by definition

$$\$n = \$E_n$$

Thus either way the final term in our equation for $\$C$ reduces to $(\$n - \$n)$, so $\$C = £\alpha$. □

Next we will consider what happens when we take estimates on Waodags which are not and-dags. In what follows we will find it convenient to talk about a particular node being part of different Waodags, and its estimated cost in each of the dags. We will use the notation $\$_\delta n$ to specify that the estimation is being done relative the the Woadag δ. It should be obvious that any topological sort of all of the nodes of δ will, when restricted to the nodes of $\alpha \in \mathcal{A}(\delta)$, be a topological sort of α. Thus from now on when we talk of a sort τ it will be of the entire dag.

Definition 10 If C is a proper cut of the dag δ, then

$$\mathcal{A}(C) = \{\alpha \mid \alpha \in \mathcal{A}(\delta) \text{ and } C \text{ is a proper cut of } \alpha\}$$

Definition 11 We extend the definition of our cost function to proper cuts C of δ.

$$£C = \min_{\alpha \in \mathcal{A}(C)} £\alpha$$

Theorem 3 *Let δ be a Waodag, and C a proper cut of δ.*

$$\$_\delta C \leq £C$$

Proof. We will prove this by induction on a sequence of Woadags $\delta_1 \ldots \delta_j$ such that $\delta_1 = \alpha$, where α is the member of $\mathcal{A}(C)$ such that $£\alpha = \min_{\alpha' \in \mathcal{A}(C)} £\alpha'$, and $\delta_j = \delta$. To go from Wodag i to $i+1$ we do one of the following operations:

1. Add a leaf node $l \in L_\delta$
2. Add an and-node $n \in N_\delta$, and E_n. This step is only allowed if the heads of all the edges E_n are already in δ_i.
3. Add an or-node n and one e_n. Only allowed if the head of e_n is already in δ_i.
4. Add an edge from an or-node already in δ_i, to one of its children already in in δ_i.

We will assume that it is clear that we can, in fact, build δ from α in this fashion.

We will now show that for all $i, 1 \leq i \leq j, \$_{\delta_i} C \leq £C$
Basis Step

$$\$_{\delta_1} C = \$_\alpha C = £\alpha = £C$$

The first equality is by the definition of δ_1, the second by Theorem 2, and the third by the definition of $\pounds C$. *Induction Step* We will show that for any of the operations used in going from δ_i to δ_{i+1} it must be the case that if $\$_{\delta_i} C \leq \pounds C$ then $\$_{\delta_{i+1}} C \leq \pounds C$. *Operation 1.* Since the new node is not connected to any node in C the estimated cost remains unchanged. *Operations 2 & 3.* The only possible connection between the operation and nodes in C is that one or more descendants of the heads of the edges in C might get extra parents. This would lower the estimated value for all of the parents. Thus the only possible change would be to lower the estimate cost of C. *Operation 4.* The or-node could have its estimated cost lowered if this child had lower estimated cost than any other child. There is no way its cost could be raised. Thus the estimated cost of C could only be lowered. □.

Now we relate the above material to best-first search of and-or dags in order to find the minimum cost proof.

Definition 12 A "topological successor function" $\mathcal{S}_\tau : 2^E \rightarrow 2^{2^E}$ for the topological sort τ of the and-or dag δ is defined as follows:

$$\mathcal{S}_\tau(E) = \begin{cases} \{(E - E^n) \cup E_n\} & \text{and-node} \\ \{(E - E^n) \cup \{e_n\} \mid e_n \in E_n\} & \text{or-node.} \end{cases}$$

The first branch applies if n is an and-node, and the second if n is an or-node. Again, n is the first node in τ such that $e^n \in E$, and E_n is non-empty. $\mathcal{S}_\tau$ is undefined if no such n exists. We define $\mathcal{S}^*_\tau : 2^E \rightarrow 2^{2^E}$ as follows: $E \in \mathcal{S}^*_\tau(E)$, and if $X \in \mathcal{S}^*_\tau(E)$, then $\mathcal{S}_\tau(X) \subset \mathcal{S}^*_\tau(E)$. Nothing else is in $\mathcal{S}^*_\tau(E)$.

Theorem 4 *If $E \in \mathcal{S}^*_\tau(\{e^{r_\delta}\})$, then E is a proper cut of δ.*

Proof. Suppose $E \in \mathcal{S}^*_\tau(\{e^{r_\delta}\})$. It must then be the case that we can apply $\mathcal{S}_\tau$, first to r_δ, then to a member of $\mathcal{S}_\tau(\{e^{r_\delta}\})$, etc to generate a sequence of edge sets $(F_0, \ldots, F_j)$, where $F_0 = \{e^{r_\delta}\}$, and $F_j = E$. We will prove that this sequence is a prefix of $\mathcal{E}^i_\tau(\{e^{r_\delta}\})$ for some $\alpha \in \mathcal{A}(\delta)$. From this and theorem 1 it follows that E is a proper cut of δ. The proof is by induction. *Basis step.* $F_0 = \{e^{r_\delta}\}$. Clearly $\{e^{r_\delta}\}$ is a prefix of $\mathcal{E}^i_\tau$ applied to any $\alpha \in \mathcal{A}(\delta)$.
Induction step. Assume the theorem is true for $(F_0, \ldots, F_i)$. We want to prove that $(F_0, \ldots, F_{i+1})$ is a prefix of $\mathcal{E}^i_\tau$ applied to some $\alpha \in \mathcal{A}(\delta)$ (in general there will be many such α). Consider what happens in going from F_i to F_{i+1}. $F_{i+1} \in \mathcal{S}_\tau(F_i)$, and

$$\mathcal{S}_\tau(F_i) = \begin{cases} \{(F_i - E^n) \cup E_n\} & \text{and-node} \\ \{(F_i - E^n) \cup \{e_n\} \mid e_n \in E_n\} & \text{or-node} \end{cases}$$

Now if n is an and-node, then any α for which F_i is a proper cut will also have F_{i+1} as a proper cut, since α must include n and E_n. Thus $\mathcal{S}_\tau(F_i) = \{\mathcal{E}_\tau(F_i)\}$, so $F_{i+1} = \mathcal{E}_\tau(F_i)$. On the other hand, suppose n is an or-node. Then F_{i+1} extends F_i by removing the predecessor edges of n and including one of the possible choices e_n at the or-node n. Now consider an α which includes all of E_i and also has, as its "choice" at n, e_n. For this α $F_{i+1} = \mathcal{E}_\tau(F_i)$. □

Theorem 5 *Let $C \in \mathcal{S}^*_\tau(\{e^{r_\delta}\})$. $\$C \leq \pounds C$.*

Proof. By Theorem 4, C is a proper cut of δ. Thus, by Theorem 3 $\$C \leq \pounds C$. □

Experimental Results

We tested our scheme on a collection of and-or dags generated by the Wimp3 natural language understanding program [2,6]. Wimp3 understands stories by generating Bayesian networks [8] to model the possible interpretations of the text, and evaluates the networks to find which interpretation is most probable given the evidence. With a small amount of post-processing we were able to convert these networks into equivalent and-or dags. We tested our heuristic on all of the dags from 15 stories of the 25-story corpus used to debug Wimp. (We were not able to use all of the stories because many of them used features in Wimp which would have made the conversion to and-or dags much more difficult. However, there is no reason to believe that this would have any effect on the comparisons between heuristics.) In all we ran the minimal-cost proof schemes on 140 dags ranging in size from 7 nodes to 168 nodes. As a method of comparison we have used time (in seconds) as a function of number of nodes. Since all of the methods were run on the same machine, using mostly the same code, the normal arguments against raw times did not seem to apply. We might also note that a comparison based upon number of partial solutions explored (the other obvious method) would, if anything, tend to make our new methods look still better since it would hide the extra time per iteration they require. Since the searches are exponential in principle, we have done a least-squares fit of the equation

$$time = e^{a+b \cdot |nodes|}.$$

The results are as follows:

Method	a	b
Cost-so-far	-6.4083	.07932
Cost sharing	-5.6498	.06337
Multiple parent removal	-5.5376	.05877

The reduction in the exponent shows that cost sharing does, in fact, pay off.

It should be clear that it is possible to improve our cost estimator. In particular, \$ will be optimistic when there are multiple parents of a node which came from a single or node. For example, in Figure 6 there are two parent edges of n which came from or. In such cases, no partial proof will have more than one of them, so prorating the cost of the child among its parent edges leads to a underestimate of the cost at parent nodes. This is okay in the sense that the heuristic is admissible, but it would be better if it could be, at least in part, corrected for. We have implemented an improved

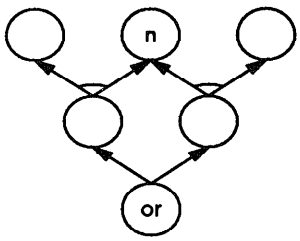

Figure 6: Two edges from the same or-node

version of our estimator which catches some of these situations. It indeed performed even better than the original cost sharing scheme as seen by the final line in the above table. We should note, however, that these data are still preliminary, and hide a lot of interesting behavior which deserve further investigation.

Conclusion

The usefulness of our improved cost estimate for minimal cost proofs depends on three factors: (1) to what degree important problems in AI can be formulated as minimal-cost proofs, (2) the reasonableness of our formalization of minimal-cost proofs as the search of already given and-or dags, and (3) the improvement provided by our cost estimate. Of these (1) is beyond the scope of this paper. The technique has proven useful so far, and we can only express our belief that it will continue to do so. Factor (3) was addressed by the last section. Here we want to consider point (2).

An alternative view of minimal-cost proofs is provided by Hobbs and Stickel [7]. They use a backward-chaining theorem prover to construct proofs and use the cost-so-far heuristic to guide the theorem prover. The problem with their model from the viewpoint of this paper is that our cost estimator cannot be used within it. We propagate the cost of the leaves down through the dag, but the theorem prover has no idea what the leaves are, and thus no costs can be propagated. Indeed, when viewed from this perspective, it might seem that our approach is hopeless. We need the complete dag before we can proceed, but we cannot use our heuristic to help construct the dag.

Fortunately, there is another perspective. Backward chaining proof procedures are exponential in the size of the dag they explore. This is because each partial proof combines the choices at or-nodes in a different way, and thus we get a combinatorial explosion. But simply constructing the dag need not be expensive. Indeed, given reasonable assumptions, the process can be linear in the size or the dag, or, at worst, linear in the size of the dag plus the knowledge base which underlies the dag. We get this improvement because the dag lays out an exponential number of proofs in a compact form. Of course, it is not possible by just looking at the dag to easily determine which proof is minimal or, even (if we allow negation), to see if there is *any* consistent proof available.

But this is okay with us. Once we have a dag (constructed in linear time), we can go back to work using our cost estimator. Or, to put this slightly differently, we can build a dag in linear time at the cost of a subsequent exponential process to find the minimal-cost proof. The contribution of this paper is to make this latter process more efficient.

References

1. CHARNIAK, E. A neat theory of marker passing. Presented at *AAAI-86* (1986).
2. CHARNIAK, E. AND GOLDMAN, R. P. A semantics for probabilistic quantifier-free first-order lanuages, with particular application to story understanding. Presented at *IJCAI-89* (1989).
3. CHARNIAK, E. AND HUSAIN, S. A New Admissible Heuristic for Minimal-Cost Proofs. Department of Computer Science, Brown Univeristy, Technical Report, Providence RI, 1991.
4. CHARNIAK, E. AND SHIMONY, S. E. Probabilistic semantics for cost based abduction. Presented at *Proceedings of the 1990 National Conference on Artificial Intelligence* (1990).
5. GENESERETH, M. R. The use of design descriptions in automated diagnosis. *Artificial Intelligence 24* (1984), 411–436.
6. GOLDMAN, R. AND CHARNIAK, E. Dynamic construction of belief networks. Presented at *Proceedings of the Conference on Uncertainty in Artificial Intelligence* (1990).
7. HOBBS, J., STICKEL, M., MARTIN, P. AND EDWARDS, D. Interpretation as abduction. Presented at *Proceedings of the 26th Annaual Meeting of the Association for Computational Linguistics* (1988).
8. PEARL, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, Los Altos, Calf., 1988.
9. REITER, R. A theory of diagnosis from first principles. *Artificial Intelligence 32* (1987), 57–95.
10. SELMAN, B. AND LEVESQUE, H. J. Abductive and default reasoning: a computational core. Presented at *Proceedings of the Eighth National Conference on Artificial Intelligence* (1990).
11. SHANAHAN, M. Prediction is deduction but explanation is abduction. Presented at *Ijcai* (1989).
12. SHIMONY, S. E. On irrelevance and partial assignments to belief networks. Computer Science Department, Brown University, Technical Report CS-90-14, 1990.
13. STICKEL, M. E. A Prolog-like inference system for computing minimum-cost abductive explanations in natural-language interpretation. SRI International, Technical Note 451, 1988.