An Analysis of the Conspiracy Numbers Algorithm

Lisa Lister

Jonathan Schaeffer

Department of Computing Science
University of Alberta
Edmonton, Alberta
Canada T6G 2H1

ABSTRACT

McAllester's Conspiracy Numbers algorithm is a minimax search procedure that builds game trees to variable depths without application-dependent knowledge. The algorithm gathers information to determine how likely it is that the search of a sub-tree will produce a useful result. "Likeliness" is measured by the conspiracy numbers, the minimum number of leaf nodes that must change their value (by being searched deeper) to cause the minimax value of a sub-tree to change. The search is controlled by the conspiracy threshold (*CT*), the minimum number of conspiracy numbers beyond which it is considered unlikely that a sub-tree's value can be changed.

This paper analyzes the best-case performance for the algorithm. McAllester's original algorithm is shown to build a tree of size $O(w^{\frac{CT}{w-1}})$, where *w* is the branching factor of the tree. A new improvement to the algorithm is shown to reduce this complexity to $O(CT^2)$. Hence for a given fixed *w*, the algorithm's best case can be improved from exponential to quadratic growth. Although the minimal tree case is not necessarily representative of the search trees built in practice, experimental results are presented that demonstrate the improvement appears to be significant in the expected case as well.

Keywords: Conspiracy numbers, alpha-beta search, minimax search, state space search, heuristic search.

1. Introduction

There are many well-known methods for efficiently searching minimax trees. *Alpha-beta* ($\alpha\beta$) [1] and *SSS** [2], for example, are elegant algorithms that greatly reduce the search effort required. However, both have a fundamental limitation: a large portion of the search effort must be devoted to the exploration of sub-trees which have a small chance of being part of the solution tree, yet must be considered to be certain the algorithm returns the correct result. As a result, recent research activity has focused on methods for concentrating the search effort in regions of the tree that are most likely to yield interesting results. Recently, a number of inventive search algorithms have appeared for building minimax trees to variable depths in an application-independent manner. These include enhancements to the basic $\alpha\beta$ framework (singular extensions [3] and null moves [4, 5]) and entirely new methods for building these trees (Min/Max Approximation [6]; Solution Tree and Costs Search (STC) [7]; Equi-Potential Search (EPS) [8]; Conspiracy Numbers [9, 10]).

McAllester's *Conspiracy Numbers* algorithm is a new approach to minimax search [9, 10]. Rather than searching to a fixed depth, the algorithm selectively expands nodes in the tree until a specified degree of confidence is achieved in the root value. Confidence is defined by a value's conspiracy number: the minimum number of leaf nodes that must change their value (or *conspire*) to cause the root of the tree to change to that value. The search is controlled by the conspiracy threshold (*CT*), the minimum number of conspiracy numbers beyond which it is considered unlikely that a sub-tree's value can be changed. The novelty of the algorithm is that it selectively expands nodes in an *application-independent* manner, without requiring, for example, extensive leaf node domain-dependent knowledge (such as *B** [11]). As the algorithm is new, there is little theoretical ([7, 9, 12-14]) and experimental ([15-18]) data on its performance.

In this paper, an analysis of the best-case performance of the conspiracy numbers algorithm is presented. The minimal search tree (relative to a given *CT*) occurs when the evaluation function always returns the same value. In this case, McAllester's original algorithm builds a tree of size $O(w^{\frac{CT}{w-1}})$, where w is the branching factor of the tree. A new improvement to the algorithm is shown to reduce this complexity to $O(CT^2)$. Hence for a given fixed w, the algorithm's best case can be improved from exponential to quadratic growth.

Best case analysis of an algorithm does not necessarily translate into improved performance in practice. Incorporating the modification suggested in this paper into a program that solves chess problems demonstrates that the improvement appears to be significant in the expected case as well.

Section 2 provides a brief overview of the conspiracy numbers algorithm. Section 3 presents an analysis that proves McAllester's original algorithm is not optimal. In Section 4, an improvement to the algorithm is shown to reduce the best case complexity from exponential to quadratic growth. Section 5 briefly relates some experiments using the new algorithm to solve chess problems. Finally, Section 6 presents the conclusions and further work.

2. Conspiracy Numbers

The following brief description of the algorithm is adapted from Klingbeil and Schaeffer [17]. More detailed descriptions have been given by McAllester [9, 10] and Schaeffer [18]. Appendix A contains a pseudo-code description of the algorithm. The algorithm and its analysis is based on McAllester's original description of the algorithm [10]. A later formulation included enhancements, such as iterative deepening and bound sequences [9], which do not affect the conclusions of this paper.

Conspiracy numbers provide a measure of the difficulty to change the current minimax value of a node. In Figure 1, assuming the root is a maximizing node, how many leaf nodes in the tree have to change their value, as a result of being searched one ply deeper, to cause the value at the root (t_{root}) to become 2? The simplest way would be if node *J*'s value changed to 2. Another way would be for both nodes *F* and *G* to change their values appropriately. Nodes *F* and *G* form a *set of conspirators* for increasing t_{root} to 2; both have to *conspire* to achieve this result. Node *J* also forms a set of conspirators for increasing t_{root} to 2, in this case the *minimal* set. The minimum number of leaf nodes that must conspire to change t_{root} to a specific value is called the *conspiracy number (CN)* for that value. Table 1 shows the conspiracy numbers for Figure 1 along with the minimal set of conspirators for each value.

It turns out that there are simple recursive relations for calculating the conspiracy numbers of a node from the conspiracy numbers of its descendents. In what follows, let m denote the minimax value of a node and v denote the value we would like to change m to.

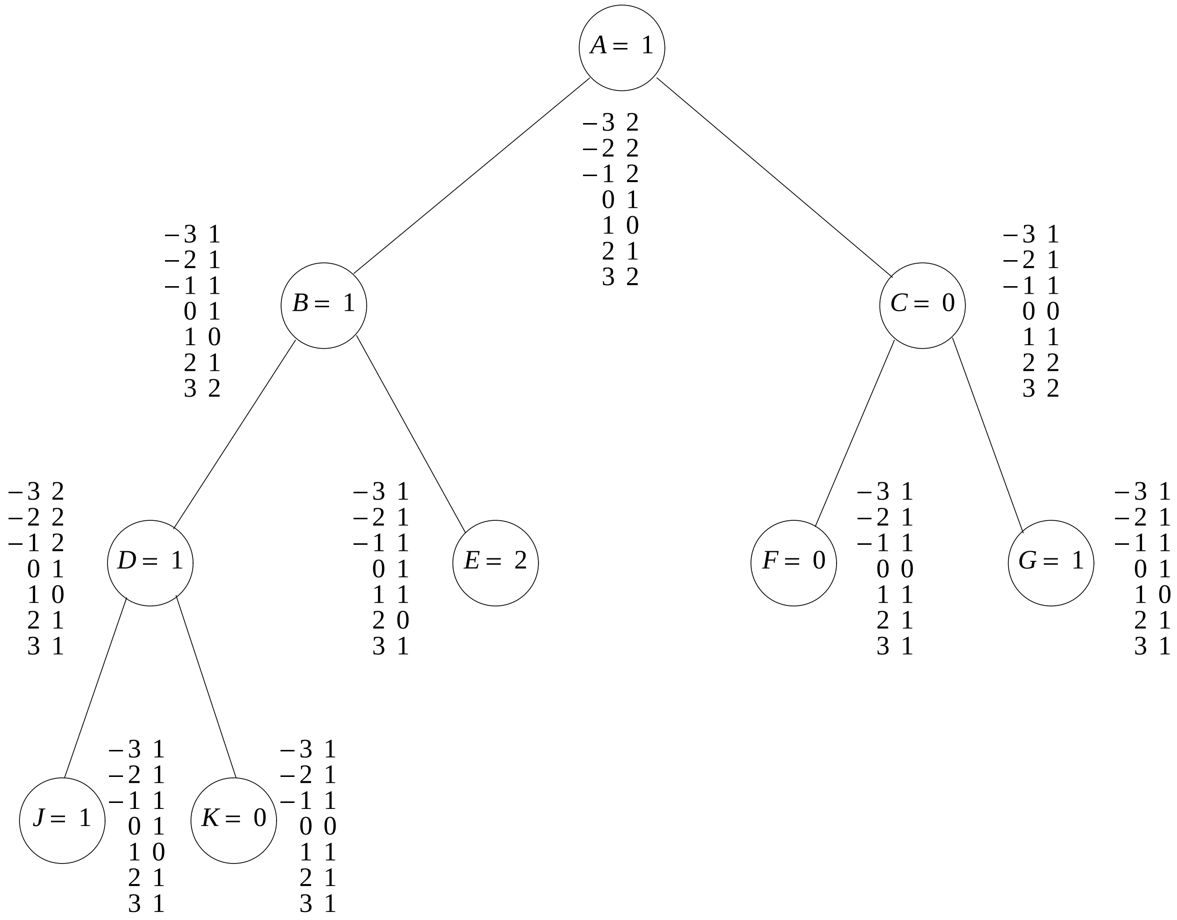

Figure 1. Conspiracy Numbers.

Value	CN	Nodes to Change
-3	2	(E and (F or G))
-2	2	(E and (F or G))
-1	2	(E and (F or G))
0	1	(E or J)
1	0	
2	1	(J or K)
3	2	(E and (J or K)) or (F and G)

Table 1. Conspirators.

At a leaf node, changing m to any other value requires a conspiracy of only that node itself, and hence has a conspiracy number of 1. If we do not want to change the node's value, then no conspiracy is required and the conspiracy number is 0. If the leaf node is also a terminal node, then there is no way to change its value and a conspiracy number of ∞ is assigned. Hence, the conspiracy numbers for a leaf node, *T*, are:

$$CN(T,\ v)\ =\ \begin{cases} 0 & \text{if}\ \ v = m \\ 1 & \text{if}\ \ v \neq m \\ \infty & \text{if}\ \ terminal\ node \end{cases}$$

At a maximizing interior node, *T*, to increase the value to *v* requires only one of the sons to change its value to *v*. Assuming that the conspiracy number for each son has already been calculated, then the minimum number of conspirators required to increase the node to *v*, $\uparrow(T,\ v)$, is just the minimum number of conspirators to increase one of the sons to *v*. This yields the following relation:

$$\uparrow(T,\ v)\ =\ \begin{cases} 0 & \text{for}\ all\ v \leq m \\ \underset{all\ sons\ i}{MIN}\ \uparrow CN(T_{i,}\ v) & \text{for}\ all\ v > m \end{cases}$$

To decrease the node's value to *v*, $\downarrow CN(T,\ v)$, requires all sons whose value is greater than *v* to decrease their value to *v*. Given the minimal set of conspirators for decreasing each son to *v*, *all* members of each of these sets must conspire together to decrease the node's value to *v*. Therefore:

$$\downarrow CN(T,\ v)\ =\ \begin{cases} 0 & \text{for}\ all\ v \geq m \\ \sum\limits_{all\ sons\ i} \downarrow CN(T_{i,}\ v) & \text{for}\ all\ v < m \end{cases}$$

For a minimizing interior node, the following dual relations apply:

$$\uparrow CN(T,\ v)\ =\ \begin{cases} 0 & \text{for}\ all\ v \leq m \\ \sum\limits_{all\ sons\ i} \uparrow CN(T_i,\ v) & \text{for}\ all\ v > m \end{cases}$$

$$\downarrow CN(T,\ v)\ =\ \begin{cases} 0 & \text{for}\ all\ v \geq m \\ \underset{all\ sons\ i}{MIN}\ \downarrow CN(T_{i,}\ v) & \text{for}\ all\ v < m \end{cases}$$

Figure 1 shows the conspiracy numbers for each node, with $\uparrow CN$ and $\downarrow CN$ merged into one vector. It is worth noting the *monotonicity* property of conspiracy numbers. If $v\ <\ w$ then $\uparrow CN(T,\ v) \leq \uparrow CN(T,\ w)$ and $\downarrow CN(T,\ w)\ \geq\ \downarrow CN(T,\ v)$. Also, given a set of conspirators for changing the value of a node to *v*, ($v \neq m$), this same set can conspire to change the node to any value between *m* and *v*.

Since conspiracy numbers represent the difficulty of changing the value of a node, one way they can be used is to judge the accuracy of the root value. A *conspiracy threshold (CT)* is introduced that specifies the minimum number of conspirators required before we consider it unlikely a node can take on that value. A value v is a likely value if $CN(T, v) < CT$. The range of likely root values is given by $[t_{\min}, t_{\max}]$, where $t_{\min} \le t_{root} \le t_{\max}$.

The algorithm continues to search until it has narrowed the range of likely values to just one value. Once all root values but one have been ruled out, we expect further search will not change that value. The higher the threshold, the greater the confidence in the final root value.

Given a range of likely root values, how do we rule out all but one of them? The obvious way is to rule them out one by one, starting with either $t_{\max}$ or $t_{\min}$. To rule out $t_{\max}$, the algorithm tries to either change the root value to $t_{\max}$ or increase the corresponding conspiracy number for $t_{\max}$ to at least CT (*IncreaseRoot*). This is done by "proving" that a member of the minimal conspiracy set will not conspire with the other members of the set to help change the value of the root node to $t_{\max}$. A similar strategy exists for ruling out $t_{\min}$ (*DecreaseRoot*).

During each step of the tree growth procedure, the algorithm must choose to either *IncreaseRoot* or *DecreaseRoot.* Faced with these two alternatives, it chooses to attempt to rule out the value which is furthest from t_{root}. If both are equidistant from the root value, it then arbitrarily chooses to *DecreaseRoot.* Having made a decision to rule out $t_{\max}$, for example, a leaf node from the minimal set of conspirators must be found to search one ply deeper (or *expanded*). To find this node, the algorithm descends from the root using the following procedure:

a) at a maximizing node

Only one successor node must increase its value to $t_{\max}$ for the parent root node to do likewise. The most likely branch is the one requiring the least number of conspirators to increase it to $t_{\max}$. After computing $CN(T, t_{max})$ for each successor, choose the successor node requiring the minimum conspirators. If more than one branch has the minimum, arbitrarily choose the left-most one.

b) at a minimizing node

Here there may be many descendent nodes that have to increase their value to increase the parent node to t_{max}. Each such branch contains conspirators which together form the set of conspirators to increase this node to t_{max}. Again the algorithm can choose to traverse any of the appropriate branches and we arbitrarily choose to take the left-most one.

Having reached a leaf node, that node is expanded (i.e. searched one ply deeper). Since each descendent may yield a favorable or unfavorable assessment, the descendents are ordered according to the results of their evaluation. By putting the more favorable descendents first, this increases the chances that the left-most descendent is the best, justifying the above choices. The minimax value and conspiracy numbers are passed back up the tree, resulting in new numbers along the path from the root to the leaf node.

What is being accomplished by expanding this node? If we are successful at increasing the value of this node to t_{max}, then the number of conspirators in this set has been decreased by one and therefore other members of the set can be expanded to see if they will conspire successfully. If the value is less than t_{max} and the expanded node is minimizing, then we may have been successful at increasing the number of conspirators at the root (i.e. increased the minimal set of conspirators). The number of conspirators may have reached *CT*, resulting in a narrowing of the range of likely values at the root. At a maximizing expanded node with a value less than t_{max}, nothing has been accomplished towards ruling out t_{max}.

A dual strategy exists for ruling out t_{min}. This tree growth procedure was McAllester's original proposal.

3. Analysis of McAllester's Algorithm

Throughout the paper, five assumptions are made:

(1) If one successor of a node is present, then all successors are evaluated. This assumption is made to be consistent with McAllester's algorithm.

(2) Every leaf node can be expanded. Including terminal nodes does not change the analysis.

(3) It is assumed that $CT > 1$. Otherwise, McAllester's algorithm converges trivially with a single node.

(4) Trees are rooted at a MAX node, unless otherwise specified.

(5) Every node has *w* successors. Generalizing this does not change the results presented here, but does

complicate the analysis.

The following notation is used throughout. For a given tree $\overline{T}$,

(1) T is the root node,

(2) $T_{1,} \ldots, T_w$ are the successors of T,

(3) $\overline{T}_{1,} \ldots, \overline{T}_w$ are the trees rooted at $T_{1,} \ldots, T_w$, respectively,

(4) t is the minimax value of T (or $\overline{T}$),

(5) $[t_{\min}, t_{\max}]$ is the range of likely values of T, and

(6) v the minimax value of tree T. Note that in this description of the algorithm all values of v are permissible. However, in a practical implementation, v is selected from a small finite set.

The term *full tree* of depth d is used to describe a tree where every interior node has w successors and all leaf nodes are d branches from the root. In the context of game-tree searching, a minimax tree is a full tree (no αβ cut-offs).

At times, the analysis becomes easier to follow by referring to the algorithm description given in the Appendix. In these places, pointers are given to blocks of pseudo-code in the Appendix. For further details on any of the proofs given in this section, the reader is referred to Lister [13].

3.1. Minimality

The minimal tree grown by McAllester's algorithm occurs when all leaf nodes evaluate to the same value (the proof is in [13]). Since the values are identical, the expansion of a node produces the same value as its parent and consequently can only cause conspiracy numbers in the tree to stay the same or increase. The more rapidly the conspiracy numbers increase, the sooner convergence will occur. If the leaf nodes do not produce identical values, then the conspiracy numbers might decrease at some nodes.

McAllester has shown that when all leaf nodes return the same value, the algorithm will produce a tree identical to that of a d–ply αβ search [9]. However, the order in which nodes are selected for expansion is not the same as in αβ. Also, the tree growth may converge for a given CT before it has constructed the αβ tree.

In the following analysis, we distinguish between a constant static evaluation function that always returns the same value (*CEF*) and the general arbitrary evaluation function (*AEF*). Unless otherwise stated, *CEF*s are assumed. The term *McAllester tree* will refer to the tree constructed by McAllester's algorithm (as given in the Appendix) when a *CEF* is used.

3.2. McAllester's Tree Growth Procedure

Examination of McAllester's algorithm shows that trees are built in three stages:

I. Build a tree for which the range of likely values is reduced from $[-\infty, +\infty]$ to $[v, +\infty]$ for some finite v. Such a tree is denoted by $\overline{P}_m$ below.

II. Expand the $\overline{P}_m$-tree to reduce the range of likely values from $[v, +\infty]$ to $[l, u]$ for some finite l and u. This tree is denoted by $\overline{C}_m$ below. For an evaluation function that always returns the same value, it will be shown that the $\overline{C}_m$-tree is convergent and the algorithm terminates.

III. For an arbitrary static evaluation function, McAllester's algorithm continues expanding the $\overline{C}_m$-tree until convergence occurs (it may not) or some other stopping criteria (such as elapsed time) is applied.

3.2.1. Analysis of Stage I

The tree produced during Stage I of McAllester's algorithm is examined. It is shown that a $\overline{P}_i$-tree, $i = 0, ..., m$ for some integer m, is always built.

Definition 1: A $\overline{P}_i$-tree is defined recursively in Figure 2, where P_i is the root node and $B_2, ..., B_w$ are leaf nodes. The node B_1 has w successors, each of which is a $\overline{P}_{i-1}$-tree. A $\overline{P}_0$-tree is defined to have w leaf nodes, $B_1, ..., B_w$.

It is easy to see that a $\overline{P}_i$-tree has a depth of $2i + 1$.

Definition 1 describes the $\overline{P}_i$-tree as a specific collection of w $\overline{P}_{i-1}$-trees. Alternatively, the $\overline{P}_i$-tree can be described (less transparently) in terms of appropriate expansions of certain leaf nodes of a $\overline{P}_{i-1}$-tree.

Lemma 1: The root of a $\overline{P}_i$-tree is also the root of a $\overline{P}_{i-1}$-tree.

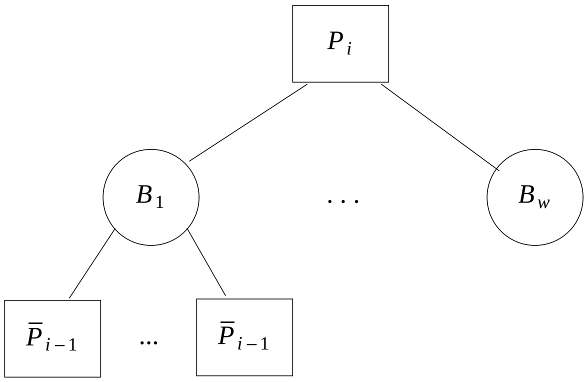

Figure 2. $\overline{P}_i$**-tree.**

Proof: For $i \geq 1$, it is asserted that the deletion of all nodes at depth $2i$ and $2i+1$ from a $\overline{P}_i$-tree gives precisely a $\overline{P}_{i-1}$-tree. The assertion is proved by induction. □

The expansion of certain leaves of the $\overline{P}_{i-1}$-tree to yield the $\overline{P}_i$-tree is exactly the construction obtained by McAllester's algorithm initially in Stage I (Theorem 1 below). It is necessary to know for which *i* McAllester's algorithm successfully completes this stage. To this end, consider $\downarrow CN(P_{i-1}, -\infty)$, the minimum number of conspirators required to decrease the minimax value of $\overline{P}_{i-1}$ to $-\infty$.

Lemma 2: For an *AEF*, let $\overline{T}$ rooted at a MAX node be a minimax tree such that

$$\overline{P}_{i-1} \subseteq \overline{T} \subset \overline{P}_i . \tag{1}$$

Then,

$$\downarrow CN(T, -\infty) = (i - 1)(w - 1) + w , \quad \text{and} \tag{2}$$

$$\uparrow CN(T, +\infty) = 1. \tag{3}$$

Proof: It is clear from Equation 1 and Definition 1 that $\overline{T}$ contains $(w - 1)$ leaf nodes at depth one. Any one of these nodes can conspire to make the minimax value of $\overline{T}$ become $+\infty$. Thus, Equation 3 follows.

To prove Equation 2, use induction on *i*. For the initial case, $i = 1$, it is required to show that if $\overline{T}$ is such that $\overline{P}_0 \subseteq \overline{T} \subset \overline{P}_1$, then $\downarrow CN(T, -\infty) = w$. There are two cases to consider, namely $\overline{T} = \overline{P}_0$ and $\overline{P}_0 \subset \overline{T} \subset \overline{P}_1$. In the former case, $\overline{T}$ is a full tree of depth one; consequently, $\downarrow CN(T, -\infty) = \sum_{j=1}^{w} 1 = w$.

In the latter case, $\overline{T}$ is described by means of Figure 3, where $\overline{T}_j$, $j = 1, ..., w$, is either a leaf node or a full tree of depth one (compare $\overline{P}_0$ and $\overline{P}_1$ trees and select $\overline{T}$ such that $\overline{P}_0 \subset \overline{T} \subset \overline{P}_1$). Furthermore, $\overline{T}_j$ is a leaf for at least one j. Let k be such that $\overline{T}_k$ is a leaf. Then, $\downarrow CN(T_k, -\infty) = 1$ and $\downarrow CN(B_1, -\infty) = \min\limits_{1 \le j \le w} \downarrow CN(T_j, -\infty) = 1$. Consequently, $\downarrow CN(T, -\infty) = \sum\limits_{j=1}^{w} \downarrow CN(B_j, -\infty) = \sum\limits_{j=1}^{w} 1 = w$. Hence, the initial case, $i = 1$, is proved.

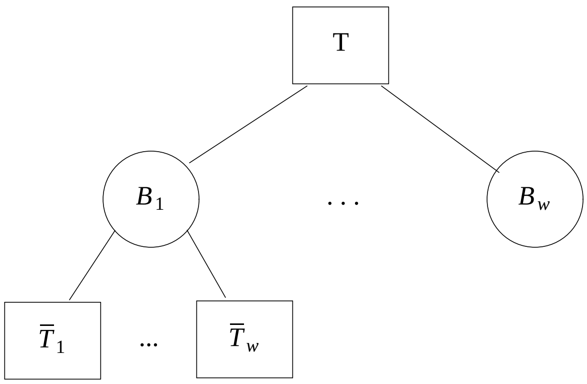

Figure 3. An intermediate $\overline{P}_1$-tree.

Inductively assume the lemma is true for $\overline{P}_{i-1} \subseteq \overline{T} \subset \overline{P}_i$. It will be shown that if $\overline{T}$ is any tree such that $\overline{P}_i \subseteq \overline{T} \subset \overline{P}_{i+1}$, then $\downarrow CN(T, -\infty) = i\,(w - 1) + w$. Observe that a tree $\overline{T}$ satisfying this relationship is given again by Figure 3, where $\overline{T}_j$ now satisfies

$$\overline{P}_{i-1} \subseteq \overline{T}_j \subseteq \overline{P}_i, \quad j = 1, ..., w, \tag{4}$$

and where, in addition for some k, $1 \le k \le w$,

$$\overline{P}_{i-1} \subseteq \overline{T}_k \subset \overline{P}_i. \tag{5}$$

From Equations 1, 2, 4, and 5, it follows that $\downarrow CN(T_j, -\infty) \ge (i - 1)(w - 1) + w$, $1 \le j \le w$, and $\downarrow CN(T_k, -\infty) = (i - 1)(w - 1) + w$. Thus,

$$\downarrow CN(B_1, -\infty) = \min_{1 \le j \le w} \downarrow CN(T_{j,} -\infty) = (i - 1)(w - 1) + w \quad , \qquad \text{and}$$

$$\begin{aligned} \downarrow CN(T, -\infty) &= \sum_{j=1}^{w} \downarrow CN(B_j, -\infty) \\ &= [(i - 1)(w - 1) + w] + \sum_{j=2}^{w} \downarrow CN(B_j, -\infty) = i\,(w - 1) + w. \ \square \end{aligned}$$

Corollary 1: For an *AEF*, let $\overline{T}$ rooted at a MIN node be a minimax tree such that $\overline{P}_{i-1} \subseteq \overline{T} \subset \overline{P}_i$. Then,

$\uparrow CN(T, +\infty) = (i - 1)(w - 1) + w$ and $\downarrow CN(T, -\infty) = 1$.

Theorem 1: In Stage I, McAllester's algorithm builds a $\overline{P}_m$-tree, where m is given by

$$m = \left\lceil \frac{CT - w}{w - 1} \right\rceil . \qquad (6)$$

The range of likely values for $\overline{P}_m$ is $[v, +\infty]$ for some finite v.

Proof: Note that since it was assumed that $CT > 1$, therefore $CT - w > 1 - w$. Thus, $\frac{CT - w}{w - 1} > -1$ and therefore $m \geq 0$. Also, note that since any finite minimax value of the root of a tree is equidistant from $-\infty$ and $+\infty$, $-\infty$ is arbitrarily chosen first for elimination from the range of likely values.

Observe that a tree consisting of a single node (the root node) cannot be convergent for $CT > 1$. Consequently, the algorithm begins by expanding the root node, yielding $\overline{P}_0$. From Equations 2 and 3, $\downarrow CN(P_{0,} -\infty) = w$ and $\uparrow CN(P_{0,} +\infty) = 1$. Thus, if $CT \leq w$, then the range of likely values for $\overline{P}_0$ is $[v, +\infty]$. The theorem is therefore true when $CT \leq w$. Equation 6 specifies the case where $m = 0$.

When $CT > w$, the range of likely values remains $[-\infty, +\infty]$ and the algorithm expands $\overline{P}_0$ (step CN1 in the Appendix). It is shown that McAllester's algorithm successively builds the sequence $\overline{P}_0, \overline{P}_1, ..., \overline{P}_m$. Assert that given $\overline{P}_{j-1}$ for some $1 \leq j \leq m$, a number of expansions are performed that eventually yields $\overline{P}_j$. More specifically (and more strongly) assert that given any tree $\overline{T}$ such that $\overline{P}_{j-1} \subseteq \overline{T} \subset \overline{P}_j$, further expansion yields a tree $\overline{T}'$ such that $\overline{P}_{j-1} \subseteq \overline{T} \subset \overline{T}' \subseteq \overline{P}_j$. Since both $\overline{P}_{j-1}$ and $\overline{P}_j$ are finite, it follows that $\overline{P}_j$ is eventually built from $\overline{P}_{j-1}$.

Proceed by induction. Assume that the assertion is true for $j = 1, ..., i$, where $i < m$. It is shown that the assertion is true for $j = i + 1$. For a tree $\overline{T}$ satisfying $\overline{P}_i \subseteq \overline{T} \subset \overline{P}_{i+1}$, from Lemma 2, $\downarrow CN(T, -\infty) = i(w-1) + w$ and $\uparrow CN(T, +\infty) = 1$. However, Equation 6 together with the assumption that $i < m$ implies that $i < \frac{CT - w}{w - 1}$. Thus, $\downarrow CN(T, -\infty) < CT$ and the range of likely values for $\overline{T}$ is $[-\infty, +\infty]$. Therefore, the algorithm is still in Stage I and the strategy selected is DecreaseRoot (step CN1 in the Appendix).

Now, examine more closely how the DecreaseRoot strategy expands $\overline{T}$. From Definition 1, $\overline{T}$ must be as in

Figure 3, where

$$\overline{P}_{i-1} \subseteq \overline{T}_j \subseteq \overline{P}_{i,} \quad 1 \le j \le w \text{ , and} \tag{7}$$

$$\overline{P}_{i-1} \subseteq \overline{T}_k \subset \overline{P}_i \tag{8}$$

for at least one k, $1 \le k \le w$. In step DR2 of McAllester's algorithm, M = $\{B_i,\ 1 \le i \le w\}$ because the minimax value $b_i > -\infty$ for all i. Therefore, B_1 (the left-most node) is selected for expansion.

Next, consider the expansion of B_1 by the DecreaseRoot strategy. Since it is a MIN node, the algorithm proceeds directly to step DR3 and expands $\overline{T}_k$, where k is the smallest integer (left-most T_j) such that $\downarrow CN(T_{k,}\ -\infty) = \min\limits_{1 \le j \le w} \downarrow CN(T_{j,}\ -\infty)$. From Lemma 2, it follows that k is the smallest integer satisfying Equation 8. By the inductive hypothesis, expansion of $\overline{T}_k$ for this k then yields a tree $\overline{T}'_k$ such that

$$\overline{P}_{i-1} \subseteq \overline{T}_k \subset \overline{T}'_k \subseteq \overline{P}_i . \tag{9}$$

Expansion of $\overline{T}_k$ into $\overline{T}'_k$ converts $\overline{T}$ in Figure 3 into a tree $\overline{T}'$. According to Equations 7, 8, and 9, $\overline{T}'$ satisfies $\overline{P}_i \subseteq \overline{T} \subset \overline{T}' \subseteq \overline{P}_{i+1}$, completing the inductive argument.

Clearly $\uparrow CN(T',\ +\infty) = 1$. Hence, $+\infty$ remains in the range of likely values. Also, observe that for $\overline{T}' = \overline{P}_m$, $\downarrow CN(P_{m,}\ -\infty) = m(w-1) + w$, by Lemma 2. Since $m = \left\lceil \frac{CT - w}{w - 1} \right\rceil$, then $\downarrow CN(P_{m,}\ -\infty) \ge CT$. Therefore, the range of likely values for $\overline{P}_m$ is [v, +∞], where v is finite. □

Theorem 1 says that successive applications of the DecreaseRoot strategy to the root, a MAX node, will yield eventually a $\overline{P}_i$-tree for any i. The dual, of course, states that successive applications of the IncreaseRoot strategy to a MIN node will yield eventually a $\overline{P}_i$-tree.

Corollary 2: Given a tree $\overline{T} \subset \overline{P}_i$ for any fixed i, application of the IncreaseRoot strategy to $\overline{T}$ whose root is a MIN node yields $\overline{T}'$ such that $\overline{T} \subset \overline{T}' \subseteq \overline{P}_i$.

3.2.2. Analysis of Stage II

After building the $\overline{P}_m$-tree (with a range of likely values $[v,\ +\infty]$) during Stage I, McAllester's algorithm continues on to Stage II. It is important to observe that later expansions performed in Stage II do not reintroduce $-\infty$ into the range of likely values.

Lemma 3: Let $\overline{T}$ be any tree with range of likely values $[v, \infty]$. Then any expansion(s) of $\overline{T}$ will not have $-\infty$ in the range of likely values.

Proof: This follows since $-\infty$ is not in $\overline{T}$'s range of likely values, and the evaluation function is finite. □

Stage I involved successive applications of the DecreaseRoot strategy. In Stage II, subsequent expansions involve only the IncreaseRoot strategy, as long as $+\infty$ remains in the range (step CN1 in the Appendix). In this stage, the $\overline{P}_m$-tree grows to become a $\overline{C}_m$-tree:

Definition 2: A $\overline{C}_i$-tree is defined recursively in Figure 4, where C_i is the root node. A $\overline{C}_0$-tree is defined to be a full tree of depth two.

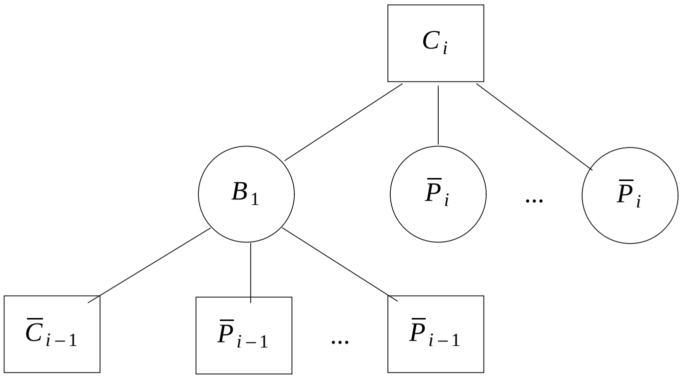

Figure 4: $\overline{C}_i$-tree.

It will be shown that it is precisely when McAllister's algorithm completes building the $\overline{C}_m$-tree, the range of likely values becomes $[l,\ u]$ for finite l and u.

Lemma 4: If $\overline{T}$ is such that

$$\overline{P}_i \subseteq \overline{T} \subseteq \overline{C}_i, \tag{10}$$

then

$$\uparrow CN(T,\ +\infty) \le i\ (w - 1) + w, \tag{11}$$

with equality only if $\overline{T} = \overline{C}_i$.

Proof: The proof is by induction. For the initial case, it is required to show that if $\overline{P}_0 \subseteq \overline{T} \subseteq \overline{C}_0$, then

$\uparrow CN(T, +\infty) \le w$ with equality only if $\overline{T} = \overline{C}_0$. A tree satisfying this condition is shown in Figure 5, where $\overline{T}_k$, $1 \le k \le w$, is a leaf node or a full tree of depth one. Thus, $\uparrow CN(T_k, +\infty) \le w$, $k = 1, ..., w$, with equality only if $\overline{T}_k$ is a full tree of depth one. Then $\uparrow CN(T, +\infty) = \min\limits_{1 \le k \le w} \uparrow CN(T_k, +\infty) \le w$ with equality only if $\overline{T}_k$, for all k, is a full tree of depth one (with equality only if $\overline{T} = \overline{C}_0$).

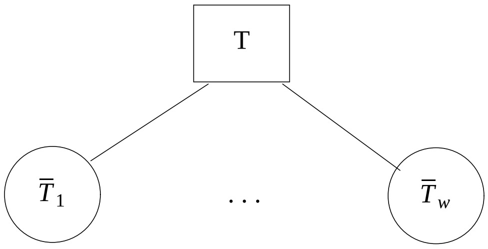

Figure 5: Intermediate $\overline{C}_0$-tree.

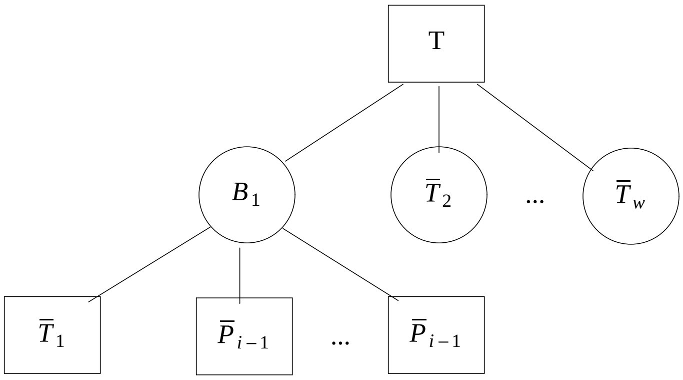

Figure 6. Intermediate $\overline{C}_{i-1}$-tree.

Inductively, assume that if $\overline{P}_{i-1} \subseteq \overline{T} \subseteq \overline{C}_{i-1}$, then $\uparrow CN(T, +\infty) \le (i-1)(w-1) + w$ with equality only if $\overline{T} = \overline{C}_{i-1}$. It is shown that for a tree satisfying Equation 10, the relationship Equation 11 holds. A tree satisfying Equation 10 can be represented by Figure 6, where $\overline{P}_{i-1} \subseteq \overline{T}_1 \subseteq \overline{C}_{i-1}$, and $B_k \subseteq \overline{T}_k \subseteq \overline{P}_i$, $2 \le k \le w$. By the inductive hypotheses, $\uparrow CN(T_1, +\infty) \le (i-1)(w-1) + w$, with equality only if $\overline{T}_1 = \overline{C}_{i-1}$. Thus,

$$\begin{aligned} \uparrow CN(B_1, +\infty) &= \uparrow CN(T_1, +\infty) + (w-1) \uparrow CN(P_{i-1}, +\infty) \\ &\le [(i-1)(w-1) + w] + (w-1) = i\,(w - 1) + w \end{aligned}$$

with equality only if $\overline{T}_1 = \overline{C}_{i-1}$. From this and the dual of Lemma 2 for MIN nodes,

$$\uparrow CN(T, +\infty) = \min \left\{ \uparrow CN(B_1, +\infty),\ \min_{2 \le k \le w} \uparrow CN(T_k, +\infty) \right\} \le i\,(w - 1) + w$$

with equality only if $\overline{T}_1 = \overline{C}_{i-1}$ and $\overline{T}_k = \overline{P}_i$, $k = 2, ..., w$ (i.e., with equality only if $\overline{T} = \overline{C}_i$). □

Finally, it is shown that McAllester's algorithm builds a $\overline{C}_m$-tree.

Theorem 2: In Stage II, a $\overline{C}_m$-tree is built from a $\overline{P}_m$-tree, where m is given by Equation 6. The range of likely values for the root of $\overline{C}_m$ is $[l, u]$ for some finite l and u.

Proof: It is shown that repeated applications of the IncreaseRoot strategy to the $\overline{P}_m$-tree of Stage I will eventually yield the $\overline{C}_m$-tree. The theorem then follows from Lemma 4 since $+\infty$ is in the range of likely values for any tree $\overline{T}$ such that $\overline{P}_m \subseteq \overline{T} \subset \overline{C}_m$, but not for $\overline{C}_m$.

Using induction, it is shown that successive applications of the IncreaseRoot strategy to $\overline{P}_j$ will yield $\overline{C}_j$. In particular, it is shown that this result is true for $j = m$, from which the validity of the theorem then follows. It is asserted that given any tree $\overline{T}$ such that $\overline{P}_j \subseteq \overline{T} \subset \overline{C}_j$, for some *j*, the next call to IncreaseRoot yields a tree $\overline{T}'$ such that

$$\overline{P}_j \subseteq \overline{T} \subset \overline{T}' \subseteq \overline{C}_j \ . \tag{12}$$

Initially, for *j* = 0, a tree satisfying these conditions is shown in Figure 5, where $\overline{T}_k$ is either a leaf node or a full tree of depth one. It must be shown that the IncreaseRoot strategy applied to $\overline{T}$ yields a tree $\overline{T}'$ satisfying Equation 12 with *j* = 0. With $\overline{T}$ so defined, the IncreaseRoot strategy proceeds until it reaches step IR3 of the algorithm (because T is a MAX node and not a leaf). Here the minimal set M is the set of all $\overline{T}_k$ which are leaf nodes (because $\uparrow CN(T_{k,}\ +\infty) = 1$ when $\overline{T}_k$ is a leaf and $\uparrow CN(T_{k,}\ +\infty) = w$ when $\overline{T}_k$ is a full tree of depth one). Consequently, the left-most leaf from the set M is expanded. This yields $\overline{T}'$ satisfying Equation 12.

Inductively, assume that the theorem is true for $j = 0, 1, ..., i - 1$. It is shown that it is true for $j = i$. If $\overline{T}$ is such that

$$\overline{P}_i \subseteq \overline{T} \subset \overline{C}_i \ , \tag{13}$$

then IncreaseRoot expands $\overline{T}$ to yield $\overline{T}'$ such that

$$\overline{P}_i \subseteq \overline{T} \subset \overline{T}' \subseteq \overline{C}_i . \tag{14}$$

For a tree $\overline{T}$ satisfying Equation 13, comparing $\overline{P}_i$ and $\overline{C}_i$ (see Figures 2 and 4), $\overline{T}$ must be the tree shown in Figure 6, where

$$\overline{P}_{i-1} \subseteq \overline{T}_1 \subseteq \overline{C}_{i-1} \tag{15}$$

and

$$B_k \subseteq \overline{T}_k \subseteq \overline{P}_i , \ 2 \le k \le w \ . \tag{16}$$

In addition, either

$$\overline{T}_1 \subset \overline{C}_{i-1}, \quad \text{or} \quad \overline{T}_k \subset \overline{P}_i \ , \quad \text{for some } k,\ 2 \le k \le w \ . \tag{17}$$

With $\overline{T}$ so defined, the IncreaseRoot strategy proceeds without effect until it reaches step IR3, as in the initial case. Here it must choose to expand one of $\overline{T}_k$, $k = 2, ..., w$, or the tree $\overline{B}_1$. The tree chosen is the one requiring the fewest conspirators to increase its minimax value to $+\infty$ (in case of a tie, the left-most is chosen). To determine which of these trees to expand, from Equation 16 and the dual of Lemma 2 for MIN nodes, observe that for $2 \le k \le w$, $\uparrow CN(T_{k,}\ +\infty) \le i(w-1) + w$ with equality iff $\overline{T}_k = \overline{P}_i$. For the remaining successor, B_1, from Equations 3 and 15, and Lemma 4,

$$\begin{aligned} \uparrow CN(B_1,\ +\infty) &= \uparrow CN(T_1,\ +\infty) + (w-1)\uparrow CN(P_{i-1},\ +\infty) \\ &\le [(i-1)(w-1) + w] + (w-1) = i\ (w-1) + w \end{aligned}$$

with equality iff $\overline{T}_1 = \overline{C}_{i-1}$. Thus, the algorithm selects to expand one of $\overline{T}_k$, $k = 2, ..., w$, or $\overline{B}_1$ for which the minimum number of conspirators is strictly less than $i\ (\ w - 1\ ) + w$. Otherwise, $\overline{T}_k = \overline{P}_i$, $k = 2, ...,$ w, and $\overline{T}_1 = \overline{C}_{i-1}$ which dictates that $\overline{T} = \overline{C}_i$, violating the inductive assumption Equation 17.

In step IR3, if the sub-tree selected for expansion is $\overline{T}_k$ for some k, $2 \le k \le w$, then the recursive call with input $\overline{T}_k$ of the IncreaseRoot strategy yields a tree $\overline{T}'_k$ such that $\overline{T}_k \subset \overline{T}'_k \subseteq \overline{P}_i$ (Corollary 2). Clearly, such an expansion of the sub-tree $\overline{T}_k$ to yield $\overline{T}'_k$ gives a tree $\overline{T}'$ satisfying Equation 14.

If the sub-tree selected for expansion is $\overline{B}_1$ (note that this can happen only if $\overline{T}_1 \subset \overline{C}_{i-1}$), consider the effect of the recursive call of IncreaseRoot with input $\overline{B}_1$ and a type of MIN node. The first step of consequence is step IR2, since the type is a MIN node. Here the successor of $\overline{B}_1$ selected for expansion is $\overline{T}_1$. Since $\overline{T}_1$ is assumed to satisfy $\overline{P}_{i-1} \subseteq \overline{T}_1 \subset \overline{C}_{i-1}$, the expansion of $\overline{T}_1$ yields a tree $\overline{T}'_1$ satisfying $\overline{P}_{i-1} \subseteq \overline{T}_1 \subset \overline{T}'_1 \subseteq \overline{C}_{i-1}$, by the inductive hypothesis. Clearly, this expansion gives a tree $\overline{T}'$ satisfying Equation 14. □

There are several important consequences arising from Theorem 2.

Corollary 3: For a *CEF*, McAllester's algorithm converges. For given *CT*, the resulting tree is the $\overline{C}_m$-tree where *m* is given by Equation 6.

Proof: First, consider any tree $\overline{T}$ such that $\overline{T} \subset \overline{C}_m$. From Lemma 4, $\uparrow CN(T,\ +\infty) < m(w-1) + w$. Using Equation 6, $\overline{T}$ is not convergent; the smallest tree produced by McAllester's algorithm that could possibly converge is the $\overline{C}_m$-tree.

It is now shown that the $\overline{C}_m$-tree does converge. Theorem 2 implies that for an *AEF* the algorithm builds a $\overline{C}_m$-tree from a $\overline{P}_m$-tree.

Now, $\downarrow CN(\overline{C}_{m,}\ -\infty)$ is required. Lemma 2 implies that for a $\overline{P}_m$-tree, $\downarrow CN(P_m,\ -\infty) = m(w - 1) + w$. Using Equation 6 and Lemma 3, $\downarrow CN(C_m,\ -\infty) \geq CT$. From Lemma 4, $\uparrow CN(C_m,\ +\infty) = m(w - 1) + w$. Using Equation 6, $\uparrow CN(C_m,\ +\infty) \geq CT$.

Finally, convergence can be shown. For any value $v < c_m$, using $\downarrow CN(C_m,\ -\infty) \geq CT$ implies that $\downarrow CN(C_m,\ v) \geq CT$ (simply change the value of the leaves of the minimal set to v instead of $-\infty$). Similarly, from $\uparrow CN(C_m,\ +\infty) \geq CT$, for any $v > c_m$, $\uparrow CN(C_m,\ v) \geq CT$. Thus, it converges. □

Using a CEF, McAllester's algorithm has built a convergent tree on completion of Stage II. With an *AEF*, the algorithm proceeds to Stage III where the $\overline{C}_m$-tree is further expanded. It is important to note that *any* tree built by McAllester's algorithm, *whether convergent or not*, must contain a $\overline{C}_m$-tree.

3.2.3. Complexity

The size of the minimal tree constructed by the algorithm will be analyzed in terms of depth and number of nodes.

Lemma 5: The depth of a $\overline{C}_m$-tree is $2m + 2$.

Proof: By induction. □

Theorem 3: The depth of a McAllester tree with conspiracy threshold *CT* is $2\left\lceil \dfrac{CT - w}{w - 1} \right\rceil + 2$.

Proof: The result is an immediate consequence of Equation 6 and Lemma 5. □

Lemma 6: The number of nodes in a $\overline{P}_m$-tree, $N(\overline{P}_m)$, is $\dfrac{(w^{m+1} - 1)(w + 1)}{w - 1}$.

Proof: A $\overline{P}_0$-tree is a full tree of depth one, hence $N(\overline{P}_0) = w + 1$. Then use induction on Definition 1. □

Lemma 7: The number of nodes in a $\overline{C}_m$-tree, $N(\overline{C}_m)$, is $\dfrac{(w+1)^2 \; w^{m+1} - 4}{w-1} - 2(m+1)w - 3$.

Proof: Since a $\overline{C}_0$-tree is a full tree of depth two, clearly $N(\overline{C}_0) = w^2 + w + 1$. Using Lemma 6 and Definition 2, the result follows by induction. □

Theorem 4: For a given *CT*, the number of nodes in a McAllester tree is

$$N_{CT} = \frac{(w+1)^2 w^{\left\lceil \frac{CT-w}{w-1} \right\rceil + 1} - 4}{w-1} - 2\left\{ \left\lceil \frac{CT-w}{w-1} \right\rceil + 1 \right\} w - 3. \tag{18}$$

Proof: The result is an immediate consequence of Equation 6 and Lemma 7. □

The results of Theorems 3 and 4 are illustrated in Table 2. For a variety of threshold (*CT*) and width (*w*) combinations, the resulting maximum depth of search (*d*) and size of search tree (N_{CT}) to solve the problem are given. An implementation of McAllester's algorithm by Norbert Klingbeil [15] has been used to verify the numbers.

	w = 10		w = 20		w = 30		w = 40	
CT	d	N_{CT}	d	N_{CT}	d	N_{CT}	d	N_{CT}
10	2	111	2	421	2	931	2	1641
20	6	13381	2	421	2	931	2	1641
30	8	134361	4	9201	2	931	2	1641
40	10	1344341	6	185561	4	29701	2	1641
50	12	13444321	6	185561	4	29701	4	68801
60	14	134444301	8	3713521	6	894541	4	68801
70	16	1344444281	8	3713521	6	894541	4	68801
80	18	13444444261	10	74273481	6	894541	6	2758321
90	20	134444444241	10	74273481	8	26841481	6	2758321
100	22	1344444444221	12	1485473441	8	26841481	6	2758321

Table 2. Depth (*d*) and number of nodes (N_{CT}) for given width (*w*) and threshold (*CT*).

Corollary 4: For a *CEF*, the number of nodes N_{CT} in a McAllester tree grows exponentially with *CT* according to $N_{CT} = O(\gamma^{CT})$, where $\gamma = w^{\frac{1}{w-1}}$ is the growth factor.

Proof: It follows from Theorem 4. □

Corollary 4 shows that the number of nodes in a tree built by McAllester's algorithm, whether convergent or not, grows exponentially with *CT*. The growth factor decreases as *w* increases.

3.3. Optimality

For a given *CT* and a *CEF*, is there a tree containing fewer nodes than a McAllester tree which converges? Here it is shown that the McAllester's algorithm is not optimal in most instances by constructing a convergent tree with fewer nodes. The tree constructed is a full tree. Some preliminary results are required.

Lemma 8: For a full tree, $\overline{F}_i$, of depth $2i$, $i \geq 0$, $\uparrow CN(F_i, +\infty) = w^i$ and $\downarrow CN(F_i, -\infty) = w^i$.

Proof: The lemma will be proved using induction. For the initial case, $i = 0$, $\overline{F}_i$ is a leaf node and the result follows trivially. Inductively assume the lemma is true for a $\overline{F}_k$-tree of depth $2k$. The $\overline{F}_{k+1}$-tree of depth $2k + 2$ is shown in Figure 7. Since T_j, $j = 1, ..., w$, is a MIN node, it follows that $\uparrow CN(T_j, +\infty) = \sum_{l=1}^{w} \uparrow CN(F_k, +\infty) = w^{k+1}$ and $\downarrow CN(T_j, -\infty) = \min_{1 \leq l \leq w} \downarrow CN(F_k, -\infty) = w^k$. Thus, $\uparrow CN(F_{k+1}, +\infty) = \min_{1 \leq j \leq w} \uparrow CN(T_j, +\infty) = w^{k+1}$ and $\downarrow CN(F_{k+1}, -\infty) = \sum_{j=1}^{w} \downarrow CN(T_j, -\infty) = w^{k+1}$. □

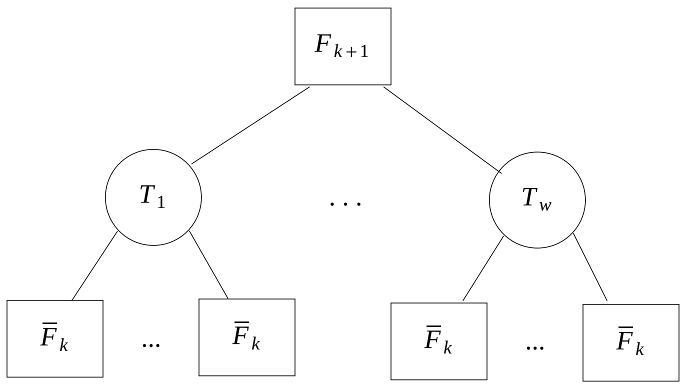

Figure 7. $\overline{F}_{k+1}$-tree.

To show that the McAllester tree is not optimal, compare the number of nodes in a McAllester tree with $N(\overline{F}_i)$. Such a comparison would be meaningless without first establishing that both trees converge for a given *CT*. For the McAllester tree, this was established in Corollary 3. For the full tree, the following corollary is required.

Corollary 5: For $CT \le w^i$, for a *CEF*, $\overline{F}_i$ of depth $2i$ is convergent.

Proof: The proof is the same as that for Corollary 3. □

Whereas the discussion in this section revolves around a *CEF*, the following result for an *AEF* is stated now for use in Section 4.

Corollary 6: For $CT \le w^i$, and for an *AEF*, the range of likely values for $\overline{F}_i$ of depth $2i$, is $[l,\ u]$ for some finite *l* and *u*.

Proof: Let f_i be the minimax value of $\overline{F}_i$. Since $\uparrow CN(F_i,\ f_i) = \downarrow CN(F_i,\ f_i) = 0$, then Lemma 8 and the monotonicity of ↑CN and ↓CN imply that the number of conspirators required to attain any finite value is finite. For $CT \le w^i$, the corollary follows. □

To show that the McAllester tree is not optimal, it is shown that there exists some tree with fewer nodes. The remaining results of this section apply only to *CEF*s.

Theorem 5: For $CT = w^i$, $i \ge 1$, there exists a convergent tree with number of nodes

$$\frac{w\ CT^2 - 1}{w - 1}. \tag{19}$$

Proof: For $CT = w^i$, construct the $\overline{F}_i$-tree. The proof is then an immediate consequence of Corollary 5 and the well-known result that a full tree of depth $2i$ has $\frac{w^{2i+1} - 1}{w - 1}$ nodes. □

Now it is possible to compare the number of nodes in the McAllester tree and $\overline{F}_i$ for specific *CT*.

Theorem 6: For $CT = w^i$, $i \ge 2$, there exists a convergent tree with strictly fewer nodes than a McAllester tree.

Proof: The McAllester tree is the $\overline{C}_m$-tree where for $CT = w^i$, $m = \left\lceil \frac{CT - w}{w - 1} \right\rceil = \frac{CT - w}{w - 1}$. According to Equation 18, the number of nodes in the $\overline{C}_m$-tree is given by

$$N(\overline{C}_m) = \frac{(w+1)^2 \, w^{\frac{CT-1}{w-1}} - 2\,w\,CT - w - 1}{w-1}. \tag{20}$$

The result follows from Theorem 5 by comparing Equations 19 with 20. Note that for $CT = w$, Equations 19 and 20 are equal. □

Rephrasing Theorem 6, yields the following corollary.

Corollary 7: For $CT = w^i$, $i \geq 2$, the McAllester tree is not optimal.

Indeed, the McAllester tree is far from optimal. In Corollary 4, it was shown that the number of nodes in a McAllester tree grows exponentially with *CT*. For the special sequence of conspiracy thresholds $CT = w^i$, $i \geq 1$, however, according to Theorem 5 there exist convergent trees for which the number of nodes grows quadratically with *CT*.

It is now known that, for $CT = w^i$, a convergent tree (the $\overline{F}_i$-tree) may be constructed with fewer nodes than the McAllester tree, except for $i = 1$ (where the number of nodes in the two trees is equal). Next, compare the two trees in general. Consider the case where $w^{i-1} < CT \leq w^i$.

For an arbitrary *CT*, the McAllester tree is the $\overline{C}_m$-tree, where $m = \left\lceil \frac{CT-w}{w-1} \right\rceil$. From Corollary 5, it follows that the $\overline{F}_i$-tree, where $i = \left\lceil \log_w CT \right\rceil$, is also convergent for a *CEF*. Comparing the number of nodes in these two trees yields the following theorem.

Theorem 7: For any $CT \geq w^3$, there exists a convergent tree with fewer nodes than the McAllester tree.

Proof: Since a full tree of depth *i* has $\frac{w^{2i+1} - 1}{w-1}$ nodes, the number of nodes in the $\overline{F}_i$-tree is given by $N(\overline{F}_i) = \frac{w^{2\lceil \log_w CT \rceil + 1} - 1}{w-1} = O(CT^2)$. The result follows by comparing this with the number of nodes in the $\overline{C}_m$-tree, for $CT \geq w^3$ (Equation 18). □

Corollary 8: There exist convergent trees where the number of nodes grows quadratically with *CT*.

Proof: The result is an immediate consequence of Theorem 7. □

For $CT < w^3$, there are cases where the McAllester tree contains fewer nodes than the smallest convergent full tree. For example, when $CT = w + 1$, the McAllester tree is $\overline{C}_1$ with $N(\overline{C}_1) = w^2(w + 3) + 1$ nodes. On the other hand, the smallest convergent full tree is $\overline{F}_2$, since $i = \left\lceil \log_w (w + 1) \right\rceil = 2$. (Note that $\overline{F}_1$ is not convergent.) $N(\overline{F}_2) = \frac{w^5 - 1}{w - 1}$. Clearly, $N(\overline{C}_1) < N(\overline{F}_2)$ for all w.

In contrast, for $CT < w^3$, there are cases where the smallest convergent tree contains fewer nodes than the McAllester tree. Let $CT = 3w - 2$, where $w \geq 3$. Since $m = 2$, the McAllester tree is $\overline{C}_2$ containing $w^4 + 3w^3 + 4w^2 - 2w + 1$ nodes. On the other hand, the smallest convergent full tree is $\overline{F}_2$ (since $i = \left\lceil \log_w CT \right\rceil = 2$), containing $w^4 + w^3 + w^2 + w + 1$ nodes. Thus, for $w \geq 3$, the full tree contains fewer nodes than the McAllester tree.

However, it is possible to do better. Let $\overline{T}$ be given as in Figure 8, with successors $T_1, ..., T_w$ and the successors of $T_4, ..., T_w$ are the same as the successors of T_3. Since $\uparrow CN(C_{0,} +\infty) = w$, $\downarrow CN(C_{0,} -\infty) = w$, $\uparrow CN(P_{0,} +\infty) = 1$, and $\downarrow CN(P_{0,} -\infty) = w$, then the following hold:

$$\uparrow CN(T_j, +\infty) = \begin{cases} 2\uparrow CN(C_0, +\infty) + (w - 2)\uparrow CN(P_0, +\infty) = 3w - 2, & j = 1, 2 \\ 2\uparrow CN(C_0, +\infty) + (w - 2) = 3w - 2, & j = 3, \ldots, w \end{cases}$$

$$\downarrow CN(T_j, -\infty) = \begin{cases} \min\left\{\downarrow CN(C_{0,} -\infty), \downarrow CN(P_0, -\infty)\right\} = w, & j = 1, 2 \\ \min\left\{\downarrow CN(C_{0,} -\infty), 1\right\} = 1, & j = 3, \ldots, w. \end{cases}$$

Thus, $\uparrow CN(T, +\infty) = \min_{1 \leq j \leq w} \uparrow CN(T_j, +\infty) = 3w - 2$ and $\downarrow CN(T, -\infty) = \sum_{j=1}^{w} \downarrow CN(T_j, -\infty) = 3w - 2$. Consequently, $\overline{T}$ is also convergent and contains $N(\overline{T}) = 2w^3 + 5w^2 - 3w + 1$ nodes, which is less than $N(\overline{F}_2)$.

In both the above examples, CT is such that $w < CT \leq w^2$. For this CT, $\overline{F}_2$ converges but $\overline{F}_1$ does not. It therefore makes sense only to compare $\overline{F}_2$ with the McAllester tree. However, it is possible that there is a tree between $\overline{F}_1$ and $\overline{F}_2$ containing fewer nodes than $\overline{F}_2$ that also converges. The two examples

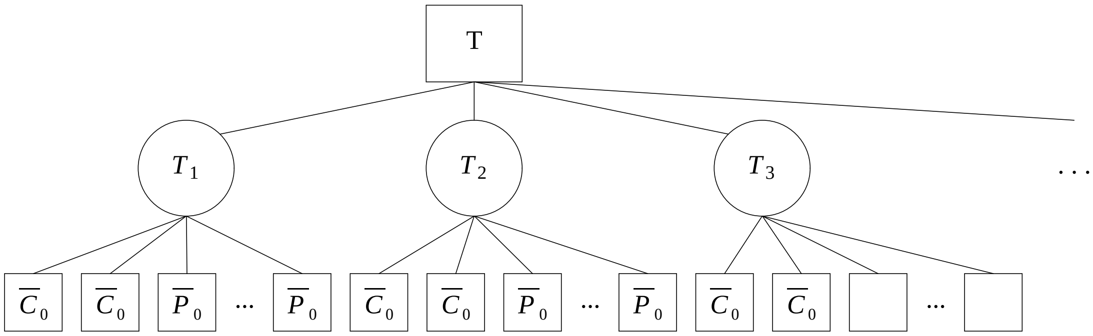

Figure 8. Intermediate full tree.

bring home the point. In the first example, $\overline{C}_1$ is such that $\overline{F}_1 \subset \overline{C}_1 \subset \overline{F}_2$, and in the second example $\overline{T}$ also satisfies $\overline{F}_1 \subset \overline{T} \subset \overline{F}_2$. It is believed that the two trees from the examples, $\overline{C}_1$ and $\overline{T}$, are optimal. For an arbitrary *CT*, a stronger statement is made.

Conjecture: For any *CT* and for a *CEF*, a convergent tree *T* with the fewest nodes satisfies $\overline{F}_{i-1} \subset \overline{T} \subseteq \overline{F}_i$, where $i = \lceil \log_w CT \rceil$. In addition, for an *AEF*, this same tree is the tree with the fewest nodes which removes $-\infty$ and $+\infty$ from the range of likely values.

As a step in this direction the following theorem is presented.

Theorem 8: For $CT \le w$, $\overline{F}_1$ of depth 2 is optimal.

Proof: Suppose $\overline{F}_1$ is not optimal. Then there exists some convergent tree $\overline{T}$ which has at least one leaf *L* at depth 1. Since $\uparrow CN(L, +\infty) = 1$, then $\uparrow CN(T, +\infty) = 1$. Hence, a contradiction exists. $\overline{T}$ is not convergent since it is assumed that $CT > 1$. □

4. Improving McAllester's Algorithm

McAllester's algorithm always grows a $\overline{C}_m$-tree (depth $2m + 2$) before it even considers the given static evaluation function. The C_m-tree is biased towards some nodes; some are expanded deeply whereas others get only a shallow expansion. For an *AEF*, McAllester's algorithm expands an exponential number of nodes in a preset pattern.

This section presents an improvement to the conspiracy numbers algorithm. As in the original algorithm, the improved version, called *Improved Conspiracy Numbers (ICN)*, begins by expanding a tree in a preset pattern. It differs from McAllester's algorithm in one major respect; in the first stages that remove $-\infty$ and $+\infty$ from the range of likely values, it produces a shallower tree with fewer nodes. Instead of growing exponentially, the number of nodes grows quadratically with *CT*.

4.1. The Improved Algorithm

Algorithm *ICN* is obtained from McAllester's by the following two modifications. In DecreaseRoot, instead of selecting $\overline{T}_j$ to be the left-most of all the successors of $\overline{T}$ in *M*, we select the left-most $\overline{T}_j$ such that

$$\downarrow CN(T_{j,}\ t_{\min}) \le \downarrow CN(T_{i,}\ t_{\min}),\ \ \text{for all}\ \overline{T}_i\ of\ M$$

(step DR2 in the Appendix). IncreaseRoot is modified in an analogous manner (step IR2).

To analyze *ICN*, further definitions and lemmas are required. Notation is required for full trees of odd depth (recall that the depth of $\overline{F}_i$ is $2i$).

Definition 3: $\overline{E}_j, j \ge 1$, is a full tree of depth $2j - 1$.

During the analysis of *ICN*, it will be necessary to know which sub-tree will be expanded next at any particular instance. This, in turn, requires knowing the number of conspirators necessary to change the minimax value of each sub-tree already produced by *ICN*. For these calculations, the following lemmas are presented.

Lemma 9: Let $\overline{T}$ be a tree such that $\overline{E}_{j-1} \subseteq \overline{T} \subset \overline{E}_j$. Then $w^{j-1} \le \downarrow CN(T,\ -\infty) < w^j$.

Proof: The proof uses induction on *j*. For brevity, the base step ($j = 2$) is omitted. Inductively, assume the assertion is true for $j = k$ and show that given any tree $\overline{T}$ such that $\overline{E}_k \subseteq \overline{T} \subset \overline{E}_{k+1}$, $w^k \le \downarrow CN(T,\ -\infty) < w^{k+1}$. Such a tree $\overline{T}$ is illustrated in Figure 9, where $\overline{F}_{k-1} \subseteq \overline{B}_p \subseteq \overline{F}_k$ (for all *p*, $1 \le p \le w$), and $\overline{F}_{k-1} \subseteq \overline{B}_q \subset \overline{F}_k$ (for some *q*, $1 \le q \le w$). Equivalently,

$$\overline{E}_{k-1} \subseteq \overline{T}_{p,\ r} \subseteq \overline{E}_{k,}\ \ \text{for all}\ p,\ r\ 1 \le p,\ r \le w,\quad \text{and} \tag{21}$$

$$\overline{E}_{k-1} \subseteq \overline{T}_{q,\ s} \subset \overline{E}_{k,}\ \ \text{for some}\ q,\ s,\ 1 \le q,\ s \le w. \tag{22}$$

If $\overline{B}_p = \overline{F}_k$, then Lemma 8 implies $\downarrow CN(B_p, -\infty) = w^k$. Since B_q is a MIN node, $\downarrow CN(B_q, -\infty) = \min_{1 \le s \le w} \downarrow CN(T_{q,s}, -\infty)$. When $\overline{B}_q \subset \overline{F}_k$, $\overline{E}_{k-1} \subseteq \overline{T}_{q,s} \subseteq \overline{E}_k$, for all q, s, $1 \le q, s \le w$ and, for some s, Equation 22 is true. Therefore, the inductive assumption implies $w^{k-1} \le \downarrow CN(B_q, -\infty) < w^k$. Since T is a MAX node, $\downarrow CN(T, -\infty) = \sum_{i=1}^{w} \downarrow CN(B_i, -\infty)$. The assertion now follows. □

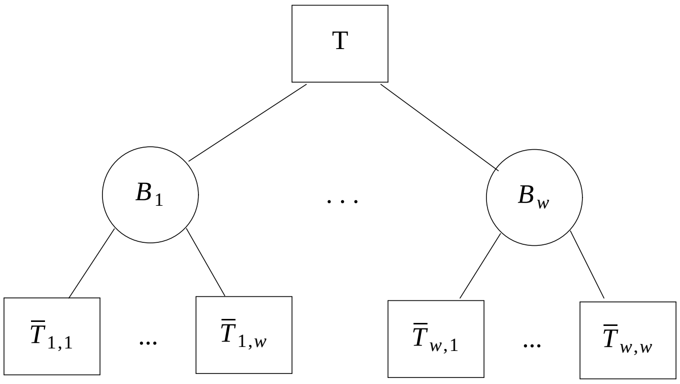

Figure 9. A $\overline{T}$-tree.

Lemma 10: Let $\overline{T}$ be a tree such that $\overline{E}_j \subseteq \overline{T} \subset \overline{F}_j$. Then $w^{j-1} \le \uparrow CN(T, +\infty) < w^j$.

Proof: The proof is similar to that for Lemma 9. □

Lemma 11: Let $\overline{F}_{j-1} \subseteq \overline{T} \subset \overline{F}_j$. Then $w^{j-1} \le \uparrow CN(T, +\infty) < w^j$.

Proof: The proof is similar to that for Lemma 9. □

The study of *ICN* is presented in order of the three stages of the conspiracy numbers algorithm. To study the first stage, the following lemma is presented.

Lemma 12: If $\overline{T}$ is such that $\overline{E}_{j-1} \subseteq \overline{T} \subset \overline{E}_j$, then the DecreaseRoot strategy in *ICN* yields $\overline{T}'$ such that $\overline{E}_{j-1} \subseteq \overline{T} \subset \overline{T}' \subseteq \overline{E}_j$.

Proof: The proof uses induction on j. For brevity, the base step ($j = 2$) is omitted. Assume that the lemma is true for $j = 2, \ldots, k$. Show that given any tree $\overline{T}$ such that $\overline{E}_k \subseteq \overline{T} \subset \overline{E}_{k+1}$, *ICN* expands $\overline{T}$ to yield $\overline{T}'$

with $\overline{E}_k \subseteq \overline{T} \subset \overline{T}' \subseteq \overline{E}_{k+1}$. To satisfy this, $\overline{T}$ must be the tree shown in Figure 9, where $\overline{E}_{k-1} \subseteq \overline{T}_{p,\,r} \subseteq \overline{E}_k$, for all $1 \le p,\ r \le w$, and $\overline{E}_{k-1} \subseteq \overline{T}_{q,\,s} \subset \overline{E}_k$, for some $q,\ s$, $1 \le q,\ s \le w$.

If $\overline{T}_{p,\,r} = \overline{E}_k$, Lemma 9 implies that $w^k \le \downarrow CN(T_{p,\,r},\ -\infty) < w^{k+1}$. When $T_{q,\,s} \subset \overline{E}_k$, Lemma 9 implies that $w^{k-1} \le \downarrow CN(T_{q,\,s},\ -\infty) < w^k$. Therefore, in the first step of consequence in DecreaseRoot, the modified version of step DR2, M is the set of nodes B_i which have one or more successors $\overline{T}_{i,s} \subset \overline{E}_k$. The left-most of these is selected; denote this to be node B_q. The subsequent call to DecreaseRoot proceeds without effect to step DR3. Now, M is the successor(s) $\overline{T}_{q,\,s}$ of B_q such that $\overline{T}_{q,\,s} \subset \overline{E}_k$. The left-most of these $\overline{T}_{q,\,s}$ is chosen for expansion and becomes $\overline{T}'_{q,\,s}$. Since $\overline{E}_{k-1} \subseteq \overline{T}_{q,\,s} \subset \overline{E}_k$, by the inductive hypothesis, after expansion $\overline{T}'_{q,\,s} \subseteq \overline{E}_k$. Therefore, $\overline{T}' \subseteq \overline{E}_{k+1}$. □

To study the second stage, the following two lemmas are presented.

Lemma 13: If $\overline{T}$ is such that $\overline{E}_j \subseteq \overline{T} \subset \overline{F}_j$, then the IncreaseRoot strategy of *ICN* yields $\overline{T}'$ such that $\overline{E}_j \subseteq \overline{T} \subset \overline{T}' \subseteq \overline{F}_j$.

Proof: The proof is similar to that for Lemma 12. □

Lemma 14: If $\overline{T}$ is such that $\overline{F}_{j-1} \subseteq \overline{T} \subset \overline{F}_j$, then the IncreaseRoot strategy in *ICN* yields $\overline{T}'$ such that $\overline{F}_{j-1} \subseteq \overline{T} \subset \overline{T}' \subseteq \overline{F}_j$.

Proof: The proof is similar to that for Lemma 12. □

Now that the preliminary results have been established, the major result of this paper is presented.

Theorem 9: Given a *CT*, let $i = \left\lceil \log_w CT \right\rceil$. Then, every convergent tree produced by *ICN* contains the $\overline{F}_{i-1}$-tree. Furthermore, for a *CEF* the convergent tree produced by *ICN* is contained within $\overline{F}_i$.

Proof: While $-\infty$ remains in the range of likely values, whenever *ICN* must select a strategy (step CN1), the DecreaseRoot strategy is chosen. The elimination of $-\infty$ will be denoted as Stage I of *ICN*.

Stage I: In this stage, it is shown that *ICN* successively produces the trees $\overline{E}_1,\ \overline{E}_2,\ \ldots$ until $-\infty$ has been eliminated from the range of likely values.

Initially, *ICN* in its first call to DecreaseRoot expands the root node producing the $\overline{E}_1$-tree (step DR1). For $i = 1,\ CT \le w$ and Stage I terminates when $\overline{E}_1$ is produced.

For $i > 1$, the range of likely values remains at $[-\infty,\ +\infty]$. Using Lemma 12, clearly *ICN* expands $\overline{E}_1$ into $\overline{E}_2$ (in a finite number of expansions), $\overline{E}_2$ into $\overline{E}_3$, ... , and so on as long as it selects the DecreaseRoot strategy (step CN1). Using Lemma 9, since $i = \left\lceil \log_w CT \right\rceil$, $-\infty$ is eliminated from the range of likely values when the algorithm has built some tree $\overline{T}$ such that

$$\overline{E}_{i-1} \subset \overline{T} \subseteq \overline{E}_i . \tag{23}$$

The elimination of $+\infty$ will be denoted Stage II of *ICN*.

Stage II: Whenever *ICN* must select a strategy (step CN1 in the appendix), the IncreaseRoot strategy is chosen. It is shown that in Stage II, *ICN* continues to expand the tree $\overline{T}$ built in Stage I yielding next a tree containing $\overline{F}_{i-1}$. At this point, $+\infty$ is still in the range of likely values. It is shown that further expansions by *ICN* of the $\overline{F}_{i-1}$-tree yield trees $\overline{T}$ satisfying $\overline{F}_{i-1} \subset \overline{T} \subseteq \overline{F}_i$. For one of these $\overline{T}$ trees, $+\infty$ will be removed from the range of likely values. Also, it is shown that for a *CEF* the convergent tree is contained within $\overline{F}_i$.

For a tree $\overline{T}$ satisfying Equation 23, it must be the tree shown in Figure 9 where $\overline{E}_{i-2} \subseteq \overline{T}_{p,\,r} \subseteq \overline{E}_{i-1}$, for all *p* and *r*, $1 \le p,\ r \le w$. Clearly either

$$\overline{E}_{i-2} \subseteq \overline{T}_{p,\,r} \subset \overline{F}_{i-2}, \quad \text{or} \tag{24}$$

$$\overline{F}_{i-2} \subseteq \overline{T}_{p,\,r} \subseteq \overline{E}_{i-1} . \tag{25}$$

If Equation 24 is true, then Lemma 10 implies that $w^{i-3} \le \uparrow CN(T_{p,\,r},\ +\infty\ ) < w^{i-2}$. If Equation 25 is true, then Lemma 11 implies that $w^{i-2} \le \uparrow CN(T_{p,\,r},\ +\infty\ ) < w^{i-1}$. If there are any trees $\overline{T}_{p,\,r}$ satisfying Equation 24, then $+\infty$ remains in the range of likely values and in the improved step IR2 of *ICN*, some tree $\overline{T}_{p,\,r}$ satisfying Equation 24 (and not Equation 25) is selected for expansion. According to Lemma 13, this expansion yields a tree $\overline{T}'_{p,\,r}$ satisfying $\overline{E}_{i-2} \subseteq \overline{T}_{p,\,r} \subset \overline{T}'_{p,\,r} \subseteq \overline{F}_{i-2}$. That is, as long as there is any sub-tree $\overline{T}_{p,\,r}$ satisfying Equation 24, *ICN* expands one such sub-tree. This expansion yields a sub-tree $\overline{T}'_{p,\,r}$ satisfying either Equations 24 or 25. Since there exist only a finite number of sub-trees satisfying

Equation 24, it follows that after a finite number of expansions, *ICN* yields a tree $\overline{T}$ for which its sub-trees $\overline{T}_{p,\,r}$ all satisfy Equation 25. At this instance, the tree $\overline{T}$ satisfies

$$\overline{F}_{i-1} \subseteq \overline{T} \subseteq \overline{E}_i . \qquad (26)$$

This means that every convergent tree produced by *ICN* contains the $\overline{F}_{i-1}$-tree.

In a way similar to Lemma 8, it may be shown that $\uparrow CN(E_i,\ +\infty) = w^{i-1}$. Thus, when $\overline{T}$ satisfies Equation 26, this implies that $+\infty$ is still in the range of likely values. Also, Equation 26 implies that $\overline{F}_{i-1} \subseteq \overline{T} \subseteq \overline{F}_i$.

According to Lemma 14, if the IncreaseRoot strategy of *ICN* is applied a finite number of times, $\overline{T} = \overline{F}_i$. Corollary 5 implies that $\overline{F}_i$ converges for a *CEF*. Thus, for a *CEF* the convergent tree produced by *ICN* is contained within $\overline{F}_i$. □

Corollary 9: For a *CEF*, the number of nodes in the convergent tree produced by *ICN* grows quadratically with *CT*.

Proof: According to Theorem 9, for a *CEF*, *ICN* produces a convergent tree contained within $\overline{F}_i$. The proof, therefore, follows directly from the proof of Corollary 8. □

Corollary 10: For an *AEF*, the number of nodes expanded by *ICN* in eliminating $+\infty$ and $-\infty$ from the range of likely values grows quadratically with *CT*.

Proof: Given a *CT*, let $i = \left\lceil \log_w CT \right\rceil$. From Theorem 9, the tree $\overline{T}$ obtained by *ICN* contains the $\overline{F}_{i-1}$-tree. For a given *AEF*, if either $-\infty$ or $+\infty$ is still in the range of likely values, the algorithm continues by expanding $\overline{T}$. As in the proof of Theorem 9, *ICN* expands a sub-tree $\overline{T}_{k,\,l} \subset \overline{F}_{i-2}$ of $\overline{T}$ and the resulting tree $\overline{T}'$ satisfies $\overline{T}' \subseteq \overline{F}_i$. From Lemma 8, $\uparrow CN(F_i,\ +\infty) = w^i$ and $\downarrow CN(F_i,\ -\infty) = w^i$. Thus, after a finite number of expansions, $+\infty$ and $-\infty$ are eliminated from the range of likely values. The result follows from $N(\overline{F}_i)$ from Theorem 7. □

The results of Theorem 9 are illustrated in Table 3. The results have been verified through experiments performed by modifying an implementation of McAllester's algorithm [15] to change it into *ICN*.

For a *CEF*, the improvement of *ICN* over McAllester's original proposal may be seen by comparing Table 3 with Table 2. For example, for $CT = 100$ and $w = 20$, McAllester's algorithm produces a tree with 1,344,444,444,221 nodes; *ICN* produces a tree with only 11,111 nodes!

	w = 10		w = 20		w = 30		w = 40	
CT	d	N_{CT}	d	N_{CT}	d	N_{CT}	d	N_{CT}
10	2	111	2	421	2	931	2	1641
20	4	2471	2	421	2	931	2	1641
30	4	3621	4	9201	2	931	2	1641
40	4	4751	4	17941	4	29701	2	1641
50	4	5861	4	17941	4	29701	4	68801
60	4	6951	4	26641	4	58411	4	68801
70	4	8021	4	26641	4	58411	4	68801
80	4	9071	4	35301	4	58411	4	135881
90	4	10101	4	35301	4	87061	4	135881
100	4	11111	4	43921	4	87061	4	135881

Table 3. ICN: Depth (*d*) and number of nodes (N_{CT}) for given width (*w*) and threshold (*CT*).

4.2. The Success of *ICN*

It is appropriate now to discuss the significance of *ICN*. Many questions arise; the most general one asks whether the new algorithm will be an improvement over McAllester's algorithm.

For a *CEF*, *ICN* is significantly better than McAllester's algorithm. Both algorithms always converge, however the number of nodes expanded by *ICN* grows quadratically with *CT*, whereas the number of nodes expanded by McAllester's algorithm grows exponentially with *CT*.

For an *AEF*, the number of nodes expanded by *ICN* grows quadratically with *CT* while eliminating $+\infty$ and $-\infty$ from the range of likely values. It has been shown that McAllester's algorithm always built a $\overline{C}_m$-tree while making the range of likely values finite. Consequently, the algorithm expands an exponential number of nodes in the same, predictable manner, regardless of the evaluation function. Thus, *ICN* looks promising.

At this time, it is not possible to state unequivocally that *ICN* expands fewer nodes than McAllester's algorithm for an *AEF*. It is possible that the initial improvement (while eliminating $+\infty$ and $-\infty$ from the range of likely values) is counteracted in the later stages of the algorithm; perhaps the *ICN* algorithm always produces a $\overline{C}_m$-tree anyway, or another tree with even more nodes than the McAllester tree. Also, there is no guarantee that *ICN* will converge, just as there was no guarantee that McAllester's algorithm would converge.

It was shown in Theorem 7 that for a *CEF* when $CT \geq w^3$, there exists a convergent full tree with fewer nodes than a McAllester tree. Under the assumptions made throughout this paper (constant branching factor w throughout and every successor of an expanded node is evaluated) it is conjectured that for *any* *CT*, *ICN* produces the tree with the fewest nodes which removes $-\infty$ and $+\infty$ from the range of likely values.

5. Experimental Results

The best case analysis of conspiracy numbers depends on the unrealistic assumption that all nodes have the same value. When that restriction is removed, it is not obvious what effect *ICN* will have on the performance.

ICN has been implemented in a program that solves chess problems [15]. On a set of 95 problems (described in [16]), the performance of *ICN* was compared with that of McAllester's original algorithm. These test positions are difficult problems, even for a human. Each problem was run for 30 minutes on a Sun 3/75 with 4 MB of memory.

It is important to emphasize that the version of Conspiracy Numbers used for the experiments is *not* the version of the algorithm analyzed in this paper. The programs used iterative deepening [9, 18], solving the problem for $CN = 2$ before moving on and solving it for a threshold of 3, then 4, then 5, etc. Iterative deepening has proven to be useful in practice, but our analysis does not include the effects of this enhancement. Another change is that the program would order sons of an expanded node using heuristic information. The use of application-dependent knowledge to order the sons, from most to least likely to succeed, improves the performance of McAllester's algorithm, because it increases the likelihood that the "best" son is in the left-most position. Without these two enhancements present, the experimental results are as one would expect: *ICN* greatly out-performs McAllester's algorithm. However, since both enhancements are common in practice, it seems more useful to compare the enhanced versions of the algorithm. Our analysis does not predict whether these changes help or hinder *ICN*'s performance. Intuitively, both should help

ICN, but to a lesser extent than for McAllester's version.

ICN solved 2 more problems than McAllester's version (40 versus 38). *ICN* required an average conspiracy number threshold of 2.47 to solve a problem, versus 2.74 for McAllester. The most significant difference was in the number of nodes expanded when the program found the solution. McAllester's variant required an average of 69,987 expansions versus 46,582 for *ICN*: 33% less.

Although the above numbers may not not appear convincing, it is important to see how they relate to the expected performance of the two algorithms. *ICN* grows quadratically with *CT*, whereas McAllester's algorithm grows exponentially. The problems that were solved were done so with relatively small *CT*s. Hence, the difference between the two algorithms should be small. As *CT* increases, however, one would expect the difference in performance between the two algorithms to grow rapidly. Unfortunately, for the problem set used, the program either solved the problem with a small threshold (2-4) or cannot solve it in the specified 30 minutes (working on threshold 8 on average).

Conspiracy numbers has been compared with the alpha-beta algorithm and shown to be a promising alternative [9, 18]. The results here suggest *ICN* may enhance these results and be a major improvement in practice. Further experimentation is required.

6. Conclusions

McAllester's conspiracy numbers algorithm is an important, new method for searching game trees. As a first step to better understanding the nature of the algorithm, a best case analysis is presented. A modification to the algorithm can reduce the growth of the search tree from exponential to quadratic. Experiments show that the new algorithm also translates to improvement in practice as well.

The conspiracy numbers algorithm gathers information from nodes in the tree that might be cut-off by $\alpha\beta$ and uses it to make decisions where best to spend search effort. This innovation has caused many to re-evaluate the utility of $\alpha\beta$ cut-offs. The result is a new set of hybrid algorithms that combine the information of conspiracy numbers and the efficiency of $\alpha\beta$ [14, 19], new algorithms that model their search expansion algorithm on the conspiracy numbers model [20], and new ways of thinking about old problems [21]. Combined with all the new, innovative minimax search algorithms that have appeared in the last few years, one can see that minimax search, once thought a "solved" problem, is now the source of many fruitful research activities.

Acknowledgments

The financial support of the Natural Sciences and Engineering Research Council of Canada, OGP 8153, is gratefully acknowledged.

References

1. D.E. Knuth and R.W. Moore, An Analysis of Alpha-Beta Pruning, *Artificial Intelligence 6*, (1975), 293-326.
2. G.C. Stockman, A Minimax Algorithm Better Than Alpha-Beta?, *Artificial Intelligence 12*, 2 (1979), 179-196.
3. T.S. Anantharaman, M.S. Campbell and F-h. Hsu, Singular Extensions: Adding Selectivity to Brute-Force Searching, *Artificial Intelligence 43*, 1 (1990), 99-110. Earlier versions of the paper appeared in the *Journal of the International Computer Chess Association*, 11(4), 135-143, 1988 and AAAI Spring Symposium, 8-13, 1988.
4. D.F. Beal, A Generalized Quiescence Search Algorithm, *Artificial Intelligence 43*, 1 (1990), 85-98. An earlier version of this paper appeared in *Advances in Computer Chess 5*, D.F. Beal (ed.), Elsevier Science Publishers, Amsterdam, 65-79.
5. G. Goetsch and M.S. Campbell, Experiments with the Null-Move Heuristic, in *Computers, Chess and Cognition*, T.A. Marsland and J. Schaeffer (ed.), Springer-Verlag, 1990. An earlier version of this paper appeared in AAAI Spring Symposium, 14-18, 1988.
6. R.L. Rivest, Game Tree Searching by Min/Max Approximation, *Artificial Intelligence 34*, 1 (1988), 77-96.
7. G. Schruffer, Minimax-Suchen: Kosten, Qualitat und Algorithmen, Ph.D. thesis, Technical University Braunschweig, 1988.
8. T.S. Anantharaman, *A Statistical Study of Selective Min-Max Search*, Ph.D. thesis, Department of Computer Science, Carnegie Mellon University, 1990.
9. D.A. McAllester, Conspiracy Numbers for Min-Max Search, *Artificial Intelligence 35*, (1988), 287-310.
10. D.A. McAllester, A New Procedure for Growing Min-Max Trees, Technical report, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1985.
11. H.J. Berliner, The B* Tree Search Algorithm: A Best First Proof Procedure, *Artificial Intelligence 12*, 1 (1979), 23-40.
12. I. Althofer, Root Evaluation Errors: How They Arise and Propagate, *Journal of the International Computer Chess Association 11*, 2 (1988), 55-63.
13. L. Lister, *Analysis of the Conspiracy Numbers Algorithm*, M.Sc thesis, Department of Computing Science, University of Alberta, 1990.
14. M. van der Meulen, Conspiracy-Number Search, *Journal of the International Computer Chess Association 13*, 1 (1990), 3-14.
15. N. Klingbeil, *Search Strategies for Conspiracy Numbers*, M.Sc thesis, Department of Computing Science, University of Alberta, 1989.
16. N. Klingbeil and J. Schaeffer, Empirical Results with Conspiracy Numbers, *Computational Intelligence 6*, (1990), 1-11.

17. N. Klingbeil and J. Schaeffer, Search Strategies for Conspiracy Numbers, *Canadian Artificial Intelligence Conference*, 1988, 133-139.

18. J. Schaeffer, Conspiracy Numbers, *Artificial Intelligence 43*, 1 (1990), 67-84. Also in *Advances in Computer Chess V*, D. Beal (ed.), Elsevier Science Publishers, Amsterdam, Netherlands, 199-218, 1989.

19. V. Allis, M. van der Meulen and H.J. van den Herik, Alpha-Beta Conspiracy-Number Search, *Advances in Computer Chess VI*, 1990, 73-96.

20. V. Allis, M. van der Meulen and H.J. van den Herik, Proof Numbers Search, Technical report 1991-5, Department of Computer Science, University of Limburg, 1991. To appear in *Artificial Intelligence*.

21. C. Elkan, Conspiracy Numbers and Caching for Searching And/Or Trees and Theorem Proving, *International Joint Conference on Artificial Intelligence*, 1989, 341-346.

Appendix. Conspiracy Numbers Algorithm

In the following description, the labels (names followed by a ':') refer to important parts of the algorithm referred to in the paper.

procedure ConspiracyNumbers(*CT*)

/* Initial tree consists of root node only */
$\overline{T}$ = Root
$t_{\min} = -\infty$
$t_{\max} = +\infty$
t = value of $\overline{T}$

/* If $t_{\min} = t_{\max}$, convergence has occurred and algorithm will terminate. Note that this condition may never be satisfied; the algorithm may continue indefinitely. */

CN1:

while($t_{\min} < t_{\max}$) **do**
  /* Select strategy. Descend tree and expand one node. */
  if(($t - t_{\min}$) < ($t_{\max} - t$)) **then**
    $[\overline{T}, t, \uparrow CN(T, v), \downarrow CN(T, v)]$ = IncreaseRoot($\overline{T}$, t_{max}, Maxnode)
  else $[\overline{T}, t, \uparrow CN(T, v), \downarrow CN(T, v)]$ = DecreaseRoot($\overline{T}$, t_{min}, Maxnode)
  end if

  /* Calculate lower and upper bound. */
  $t_{\min} = \min_{all\ v} \{v: \downarrow CN(T, v) < CT\}$
  $t_{\max} = \max_{all\ v} \{v: \uparrow CN(T, v) < CT\}$
end while
end procedure ConspiracyNumbers

function DecreaseRoot($\overline{\mathbf{T}}$, t_{min}, *type*)

/* The subroutine DecreaseRoot attempts to eliminate $t_{\min}$ from the range of likely values of $\overline{T}$. DecreaseRoot expands one leaf node of $\overline{T}$. This expanded tree is returned as the new tree $\overline{T}$. Note that *type* refers to whether the root of $\overline{T}$ is a Maxnode or a Minnode. */

DR1:

if($\overline{T}$ = terminal node) **then**

```
        ↑CN(T, t) = 0
        ↑CN(T, v) = ∞, for all v, v ≠ t
        ↓CN(T, t) = 0
        ↓CN(T, v) = ∞, for all v, v ≠ t
    end if
    if( T̄ ≠ terminal node and T̄ = leaf node ) then
        /* Expand the w successors of T̄. */
        for 1 ≤ i ≤ w do
            determine t_i
            ↑CN(T_i, v) = 1, for all v, v > t_i
            ↑CN(T_i, v) = 0, for all v, v ≤ t_i
            ↓CN(T_i, v) = 1, for all v, v < t_i
            ↓CN(T_i, v) = 0, for all v, v ≥ t_i
        end for
    end if

DR2:
    /* Non-leaf which is a Maxnode. */
    if( T̄ ≠ leaf node and type = Maxnode ) then
        /* Determine minimal set. */
        M = T̄_i, for all i, t_i > t_min
        select T̄_j to be the left-most of all the successors of T̄ in M
        /* Descend left-most sub-tree. */
        [T̄_j, t_j, ↑CN(T_j, v), ↓CN(T_j, v)] = DecreaseRoot( T̄_j, t_min, Minnode)
    end if

DR3:
    /* Non-leaf which is a Minnode. */
    if( T̄ ≠ leaf node and type = Minnode ) then
        /* Determine minimal set. */
        M = T̄_i, for all i, ↓CN(T_i, t_min) ≤ ↓CN(T_k, t_min), 1 ≤ k ≤ w
        select T̄_j to be the left-most successor of T̄ in M
        /* Descend left-most sub-tree. */
        [T̄_j, t_j, ↑CN(T_j, v), ↓CN(T_j, v)] = DecreaseRoot( T̄_j, t_min, Maxnode)
    end if

    /* Recompute ↑CN(T, v) and ↓CN(T, v). */
    if( type = Maxnode ) then
        ↑CN(T, v) = min_{1≤i≤w} ↑CN(T_i, v), for all v, v > t
        ↑CN(T, v) = 0, for all v, v ≤ t
        ↓CN(T, v) = ∑_{1≤i≤w} ↓CN(T_i, v), for all v, v < t
        ↓CN(T, v) = 0, for all v, v ≥ t
    else
        ↓CN(T, v) = min_{1≤i≤w} ↓CN(T_i, v), for all v, v < t
        ↓CN(T, v) = 0, for all v, v ≥ t
        ↑CN(T, v) = ∑_{1≤i≤w} ↑CN(T_i, v), for all v, v > t
        ↑CN(T, v) = 0, for all v, v ≤ t
    end if

    /* Recompute t. */
    if( type = Maxnode ) then
        t = max_{1≤i≤w} t_i
```

else $t = \min_{1 \le i \le w} t_i$
end if

/* Return multiple values. */
return[$\overline{T}$, t, $\uparrow CN(T, v)$, $\downarrow CN(T, v)$]
end function DecreaseRoot

function IncreaseRoot($\overline{T}$, $v_{\max}$, *type*)

/* The subroutine IncreaseRoot attempts to eliminate $t_{\max}$ from the range of likely values of $\overline{T}$. IncreaseRoot is the dual of DecreaseRoot. */

IR1:
if($\overline{T}$ = terminal node) **then**
    $\uparrow CN(T, t) = 0$
    $\uparrow CN(T, v) = \infty$, **for all** v, $v \ne t$
    $\downarrow CN(T, t) = 0$
    $\downarrow CN(T, v) = \infty$, **for all** v, $v \ne t$
end if
if($\overline{T} \ne$ terminal node **and** $\overline{T}$ = leaf node) **then**
    /* Expand the w successors of $\overline{T}$. */
    for $1 \le i \le w$ **do**
        determine t_i
        $\uparrow CN(T_i, v) = 1$, **for all** v, $v > t_i$
        $\uparrow CN(T_i, v) = 0$, **for all** v, $v \le t_i$
        $\downarrow CN(T_i, v) = 1$, **for all** v, $v < t_i$
        $\downarrow CN(T_i, v) = 0$, **for all** v, $v \ge t_i$
    end for
end if

IR2:
/* Non-leaf which is Minnode */
if($\overline{T} \ne$ leaf node **and** *type* = Minnode) **then**
    /* Determine minimal set. */
    $M = \overline{T}_i$, **for all** i, $t_i < t_{\max}$
    select $\overline{T}_j$ to be the left-most of all the successors of $\overline{T}$ in M
    /* Descend left-most sub-tree. */
    [$\overline{T}_j$, t_j, $\uparrow CN(T_j, v)$, $\downarrow CN(T_j, v)$] = IncreaseRoot($\overline{T}_j$, t_{max}, Maxnode)
end if

IR3:
/* Non-leaf which is Maxnode. */
if($\overline{T} \ne$ leaf node **and** *type* = Maxnode) **then**
    /* Determine minimal set. */
    $M = \overline{T}_i$, **for all** i, $\uparrow CN(T_i, t_{\max}) \le \uparrow CN(T_k, t_{max})$, $1 \le k \le w$
    select $\overline{T}_j$ to be left-most successor of $\overline{T}$ in M
    /* Descend left-most sub-tree. */
    [$\overline{T}_j$, t_j, $\uparrow CN(T_j, v)$, $\downarrow CN(T_j, v)$] = IncreaseRoot($\overline{T}_j$, t_{max}, Minnode)
end if

/* Recompute $\uparrow CN(T, v)$ and $\downarrow CN(T, v)$. */
if(*type* = Maxnode) **then**

$\uparrow CN(T,\ v) = \min_{1 \le i \le w} \uparrow CN(T_{i,}\ v)$, **for all** $v, v > t$
$\uparrow CN(T,\ v) = 0$, **for all** $v, v \le t$
$\downarrow CN(T,\ v) = \sum_{1 \le i \le w} \downarrow CN(T_{i,}\ v)$, **for all** $v, v < t$
$\downarrow CN(T,\ v) = 0$, **for all** $v, v \ge t$
else $\downarrow CN(T,\ v) = \min_{1 \le i \le w} \downarrow CN(T_{i,}\ v)$, **for all** $v, v < t$
$\downarrow CN(T,\ v) = 0$, **for all** $v, v \ge t$
$\uparrow CN(T,\ v) = \sum_{1 \le i \le w} \uparrow CN(T_{i,}\ v)$, **for all** $v, v > t$
$\uparrow CN(T,\ v) = 0$, **for all** $v, v \le t$
end if

/* Recompute *t*. */
if(*type* = Maxnode) **then**
$t = \max_{1 \le i \le w} t_i$
else $t = \min_{1 \le i \le w} t_i$
end if

/* Return multiple values. */
return[$\overline{T}$**,** t**,** $\uparrow CN(T,\ v)$**,** $\downarrow CN(T,\ v)$**]**
end function IncreaseRoot