Speeding up the Convergence of Real-Time Search: Empirical Setup and Proofs

David Furcy and Sven Koenig
{dfurcy,skoenig}@cc.gatech.edu

March 2000
GIT-COGSCI-2000/01

College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280

Abstract

This technical report contains the formal proofs for all of our theoretical results, as well as a description of our experimental setup for all of the results given in our AAAI-2000 paper entitled *Speeding up the Convergence of Real-Time Search*. In that paper, we propose to speed up the convergence of real-time search methods such as LRTA*. We show that LRTA* often converges significantly faster when it breaks ties towards successors with smallest f-values (*à la* A*) and even faster when it moves to successors with smallest f-values instead of only breaking ties in favor of them. FALCONS, our novel real-time search method, uses a sophisticated implementation of this successor-selection rule and thus selects successors very differently from LRTA*, which always minimizes the estimated cost to go. Our approach opens up new avenues of research for the design of novel successor-selection rules that speed up the convergence of both real-time search methods and reinforcement-learning methods. Indeed, our AAAI-2000 paper presents experiments in which FALCONS finds a shortest path up to sixty percent faster than LRTA* in terms of action executions and up to seventy percent faster in terms of trials. In this report, we first describe our experimental setup and then prove that FALCONS terminates and converges to a shortest path.

1. $s := s_{start}$.

2. $s' := \arg\min_{s'' \in succ(s)} f(s'')$,
where $f(s'') := \max(g(s'') + h(s''), h(s_{start}))$. **[F-CALC]**
Break ties in favor of a successor s' with the smallest value of $c(s, s') + h(s')$. **[TB]**
Break remaining ties arbitrarily (but systematically[1]). **[TB2]**

3. $g(s) :=$ if $s = s_{start}$ then $g(s)$ **[G-UPDATE]**
else $\max(g(s),$
$\min_{s'' \in pred(s)}(g(s'') + c(s'', s)),$
$\max_{s'' \in succ(s)}(g(s'') - c(s, s'')))$.

$h(s) :=$ if $s = s_{goal}$ then $h(s)$ **[H-UPDATE]**
else $\max(h(s),$
$\min_{s'' \in succ(s)}(c(s, s'') + h(s'')),$
$\max_{s'' \in pred(s)}(h(s'') - c(s'', s)))$.

4. If $s = s_{goal}$, then stop successfully.

5. $s := s'$.

6. Go to 2.

Figure 1: FALCONS

1 Introduction

This technical report contains two parts. In the first one, we describe our experimental setup used to obtain all of the results listed in (Furcy & Koenig 2000), as well as the domains and heuristics we tested FALCONS on. In the second part, we provide formal proofs for all of the theoretical results stated in the paper.

Let us briefly describe our new algorithm. FALCONS (see Figure 1), like LRTA*, is a real-time search algorithm that maintains heuristic values for all visited states and interleaves planning (via local searches) and plan execution. One iteration of FALCONS (Steps 2 to 6) consists of selecting the best successor state based on the heuristic values (Step 2), updating the heuristic values of the current state (Step 3), and moving to the selected successor (Step 5). These iterations are executed until FALCONS reaches the goal state. The last iteration ends at Step 4.

2 Experimental Setup

2.1 Domains and Heuristics

This section describes the domains and heuristic values we used for the experiments reported in (Furcy & Koenig 2000). In addition to the following domain-dependent heuristic values, we also experimented in all domains with the constant function Zero (Z). Note that all of our domains share the following two properties: (1) they are undirected, which means that for every action leading from state s to state s' with cost c, there is a reverse action from s' to s with cost c, and (2) they have uniform costs, which means that all action costs are one.

The **8-Puzzle** domain (Korf 1990) consists of eight tiles (numbered one through eight) in a 3x3 grid, leaving one position blank. A move is performed by sliding one of the tiles adjacent to the blank into the blank position. Since tiles are not allowed to move diagonally, the number of possible moves in each configuration is at most four: up, right, down or left. The goal state is the configuration with the blank in the center and the tiles positioned in increasing order, starting at the upper left corner and proceeding in a clockwise fashion. We used 1000 randomly selected start states among those from which the goal is reachable. In this domain, we experimented with the Manhattan distance (the sum, for all tiles, of their horizontal and vertical distances from their respective goal positions), abbreviated M, and the "Tiles Out Of Order" heuristic (the number of misplaced tiles), abbreviated T.

[1] Systematic tie-breaking is defined in Section 2.2.

For the **Gridworld** domain (Ishida 1997), we used a set of 20x20 grids in which 35 percent of the 20^2 grid cells were randomly selected as untraversable obstacles. For each grid, the start and goal positions were chosen randomly, while making sure that the goal was reachable from the start. Since we allowed moves to any of the traversable neighboring locations (including diagonal moves), we modified the Manhattan distance heuristic to be the sum, over all tiles, of the maximum of the tile's horizontal and vertical distances to its goal position. This heuristic was abbreviated *N*.

In the **Permute-7** domain (Holte *et al.* 1994), a state is a permutation of the integers 1 through 7. Therefore, the state space has $7! = 5040$ states. There are 6 operators. Each operator Op_k ($k = 2, \ldots, 7$) is applicable in all states and reverses the order of the first k integers in the state it is executed in. For example, the execution of Op_4 in state 7654321 leads to state 4567321. The goal state is 1234567. The *adjacency heuristic* (abbreviated A) computes for each state s the number of pairs of adjacent digits in the goal state that are not adjacent in s. For instance, $A(7321645) = 3$ since exactly three pairs are adjacent in the goal but not in s, namely $(3, 4)$, $(5, 6)$ and $(6, 7)$. We experimented with all 5040 states as start state.

We also used a version of the **Tower of Hanoi** domain (Holte *et al.* 1994) with 7 disks and 3 pegs. In the goal state, all disks are on the same peg, say peg number three. We experimented with 1000 randomly chosen start states. The D heuristic simply counts the number of disks that are not on the goal peg.

The **Words** domain (Holte *et al.* 1996) is a connected graph whose 4493 nodes are 5-letter English words that are pairwise connected if they differ in exactly one letter. The goal state is the word "goals". We experimented with 1000 randomly chosen start states. The L heuristic computes the number of positions (between 1 and 5) for which the letter is different from the letter at the same position in the goal state.

In the **Arrow** domain (Korf 1980), a state is an ordered list of 12 arrows. Each arrow can either point up or down. There are 11 operators that can each invert a pair of adjacent arrows. The goal state has all arrows pointing up. We experimented with 1000 randomly chosen start states among those from which the goal is reachable. The F heuristic returns the largest integer that is not larger than the number of arrows that need to be flipped divided by two.

2.2 Setup

A *trial* is a sequence of iterations during which FALCONS starts in s_{start} and reaches s_{goal} for the first time. Theorem 1 proves that each trial of FALCONS is guaranteed to terminate (see Section 3.4). At the end of each trial, FALCONS is reset into s_{start} and a new trial is executed using the heuristic values resulting from the previous trial. A sequence of trials from the first one, that uses the initial heuristic values, to the first one during which no heuristic value is modified, is called a *run*. Lemma 7 proves that each run of FALCONS is guaranteed to converge, i.e. to always follow the same path from some time on (see Section 3.4). Furthermore, Theorem 3 proves that, at the end of each run, FALCONS has converged to a shortest path (see Section 3.4).

In order for FALCONS to converge to a unique path, the secondary tie-breaking criterion (TB2) must be systematic. A tie-breaking criterion is *systematic* if, whenever a tie exists among a given set of successors of the current state, the same successor state is chosen as next state. We enforced systematicity of TB2 by (1) choosing an arbitrary ordering for the successors of each state and (2) breaking remaining ties (after TB had been applied) according to that ordering. The ordering was selected randomly at the beginning of a run and did not change during the run.

Since our main evaluation criterion is the travel cost to convergence, and since all actions have cost one, we are mainly interested in the number of actions to convergence or, in other words, the duration of a run. An *experiment* refers to a sequence of n runs of an algorithm in one domain with a given set of heuristic values. To attain statistical significance, we averaged our results over $n = 1000$ runs, except in the Permute-7 domain for which each experiment consisted of $7! = 5040$ runs, one for each possible start state. In general, the n runs of an experiment only differed from the other runs in the same experiment in two respects: (1) the start state, and (2) the random ordering selected at the beginning of each run to be used for systematic tie-breaking in TB2. In addition, in the Gridworld domain, each run used a different grid and goal state.

There are two advantages to using systematic tie-breaking. First, it ensures that FALCONS will converge to a unique path. If tie-breaking is not systematic, then FALCONS may not converge to a unique path. Instead, it may converge to a set of shortest paths and randomly switch between them after the heuristic values have converged, just like LRTA* in (Korf 1990). Systematic tie-breaking thus facilitated the detection of convergence, which happens when no heuristic value changes in the course of a run.

Second, systematic tie-breaking allowed us to carefully control our experimental conditions. In particular, we compared pairs of experiments that only differed in the algorithm tested (for example, FALCONS versus LRTA*). We only compared pairs of experiments in the same domain and with the same heuristic values. In addition, we used the same (random) ordering of successor states for systematic tie-breaking in all pairs of runs to be compared. In other words, when comparing algorithm 1 with algorithm 2, run 1 of both experiments used the same ordering, run 2 of both experiments used the same ordering (but different from that of run 1) etc... Furthermore, each pair of corresponding runs used the same start state (and the same grid and goal state in the Gridworld domain). Now, assume that we wanted to compare the travel cost to convergence of FALCONS in a particular domain and with a particular set of heuristic values (experiment 1) with that of LRTA* in the same domain and with the same set of heuristic values (experiment 2). Our experimental setup guaranteed that the only difference between run i ($i = 1, \ldots, n$) of experiment 1 and run i of experiment 2 was the algorithm tested, whereas each run was made under different conditions (namely, start state and ordering of successor states) from all of the other runs in the same experiment. This setup enabled us to test our results for statistical significance using the (non-parametric) sign test, as reported in (Furcy & Koenig 2000).

3 Theoretical Results

3.1 Definitions

S denotes the finite state space; $s_{start} \in S$ denotes the start state; and $s_{goal} \in S$ denotes the goal state.[2] $succ(s) \subseteq S$ denotes the set of successors of state s, and $pred(s) \subseteq S$ denotes the set of its predecessors. $c(s, s')$ denotes the cost of moving from state s to successor $s' \in succ(s)$. The goal distance $gd(s)$ of state s is the cost of a shortest path from state s to the goal, and the start distance $sd(s)$ of state s is the cost of a shortest path from the start to state s. Each state s has a g-value and an h-value associated with it, two concepts known from A* search (Nilsson 1971). We use the notation $g(s)/h(s)$ to denote these values. The h-value of state s denotes an estimate of its true goal distance $h^*(s) := gd(s)$. Similarly, the g-value of state s denotes an estimate of its true start distance $g^*(s) := sd(s)$. Finally, the f-value of state s denotes an estimate of the cost $f^*(s) := g^*(s) + h^*(s)$ of a shortest path from the start to the goal through state s.

D1 *G-values are admissible* iff $0 \leq g(s) \leq sd(s)$ for all states s.

D2 *H-values are admissible* iff $0 \leq h(s) \leq gd(s)$ for all states s.

D3 *G-values are consistent* iff $g(s_{start}) = 0$ and $0 \leq g(s') \leq g(s) + c(s, s')$ for all states s with $s \neq s_{start}$ and $s' \in succ(s)$, that is, if they satisfy the triangle inequality.

D4 *H-values are consistent* iff $h(s_{goal}) = 0$ and $0 \leq h(s) \leq c(s, s') + h(s')$ for all states s with $s \neq s_{goal}$ and $s' \in succ(s)$, that is, if they satisfy the triangle inequality.

D5 The state space S is *safely explorable* iff the goal distances of all states are finite

3.2 Notation

Superscripts of f-, g-, and h-values. In the following proofs, $f^t(s)$ (resp. $g^t(s)$ and $h^t(s)$) refers to the f-value (resp. g-value and h-value) of state s before the $t + 1$ value update, i.e. before Step 3 (Figure 1) of iteration $t + 1$. Thus, $g^0(s)$ and $h^0(s)$ are the initial g- and h-values of state s before Step 3 of iteration 1.

Subscripts of state variables. s_t refers to the current state before Step 5 (Figure 1) of iteration $t + 1$. Thus, $s_0 = s_{start}$.

[2] Although we assumed in (Furcy & Koenig 2000) that there was only one goal state, all of our results continue to hold in domains with multiple goals.

3.3 Assumptions

Our results hold under the following assumptions:

A1 The state space S is finite.

A2 The state space S is safely explorable.

A3 All actions costs are positive.

A4 The initial g- and h- values are admissible.

A5 The initial g-values are consistent.

Assumption A5 is only used for results pertaining to the use of FALCONS without G-UPDATE. Furthermore, A5 implies the part of A4 that pertains to the g-values, since the consistency of the g-values implies their admissibility. In practice, most admissible heuristic values are also consistent. Indeed, all of the heuristic values described in Section 2.1 are consistent.

3.4 Proofs

We first prove some lemmata pertaining to properties of the g-, h-, and f-values that are guaranteed to hold during the execution of FALCONS. Then, we prove that each trial of FALCONS is guaranteed to terminate (Theorems 1 and 2), that each run of FALCONS is also guaranteed to terminate, i.e. FALCONS always converges to a unique path (Lemma 6 and Corollary 6), and finally that the path FALCONS converges to at the end of each run is a minimum-cost path (Theorems 3 and 4). When appropriate, the following lemmata and theorems are accompanied by corrolaries that extend the results to FALCONS without the G-UPDATE rule.

Lemma 1

1. *Under assumptions A1-4, FALCONS cannot decrease the g-values.*
2. *Under assumptions A1-4, FALCONS cannot decrease the h-values.*

Proof:
Since only Step 3 of FALCONS modifies the heuristic values, we need only consider that step. Let $t \in \{1, 2, 3, \ldots\}$ be the number of the current iteration. Let s be any state in S.

1. Proof for G-UPDATE

Case (i): $s = s_t$
If $s = s_{start}$ then $g^{t+1}(s) \stackrel{G-UPDATE}{=} g^t(s)$, else

$$g^{t+1}(s) \stackrel{G-UPDATE}{=} max \left\{ \begin{array}{l} g^t(s), \\ \min_{s'' \in pred(s)} (g^t(s'') + c(s'', s)), \\ \max_{s'' \in succ(s)} (g^t(s'') - c(s, s'')) \end{array} \right\} \stackrel{\text{def. of max}}{\geq} g^t(s).$$

Case (ii): $s \neq s_t$
In this case, $g(s)$ is not updated, and thus $g^{t+1}(s) = g^t(s)$.

Therefore in both cases, $\forall t \in \{1, 2, 3, \ldots\}, s \in S$: $g^{t+1}(s) \geq g^t(s)$.

2. Proof for H-UPDATE

Case (i): $s = s_t$
If $s = s_{goal}$ then $h^{t+1}(s) \stackrel{H-UPDATE}{=} h^t(s)$, else

$$h^{t+1}(s) \stackrel{H-UPDATE}{=} max \left\{ \begin{array}{l} h^t(s), \\ \min_{s'' \in succ(s)} (c(s, s'') + h^t(s'')), \\ \max_{s'' \in pred(s)} (h^t(s'') - c(s'', s)) \end{array} \right\} \stackrel{\text{def. of max}}{\geq} h^t(s).$$

Case (ii): $s \neq s_t$
In this case, $h(s)$ is not updated, and thus $h^{t+1}(s) = h^t(s)$.

Therefore in both cases, $\forall t \in \{1, 2, 3, \ldots\}, s \in S$: $h^{t+1}(s) \geq h^t(s)$. ■

Corollary 1

1. *Under assumptions A1-5, FALCONS without G-UPDATE cannot decrease the g-values.*
2. *Under assumptions A1-5, FALCONS without G-UPDATE cannot decrease the h-values.*

Proof:
1. Since G-UPDATE is the only place in FALCONS where the g-values are updated, FALCONS without G-UPDATE never modifies the g-values and thus cannot increase them.
2. The proof is the same as that for Lemma 1(2). ■

Let us now define the start distance $sd(s)$ and goal distance $gd(s)$ of state s:

$$sd(s) := \begin{cases} 0 & \text{if } s = s_{start} \\ \min_{s' \in pred(s)}(sd(s') + c(s', s)) & \text{otherwise} \end{cases} \tag{1}$$

$$gd(s) := \begin{cases} 0 & \text{if } s = s_{goal} \\ \min_{s' \in succ(s)}(c(s, s') + gd(s')) & \text{otherwise} \end{cases} \tag{2}$$

Lemma 2

1. *Under assumptions A1-4, the g-values remain admissible during the execution of FALCONS.*
2. *Under assumptions A1-4, the h-values remain admissible during the execution of FALCONS.*

Proof:
1. Proof by induction on t.
At $t = 0$, assumption A4 guarantees that $\forall s \in S$: $g^0(s)$ is admissible.
Assume that the induction hypothesis holds at the beginning of iteration t:

$$\forall s \in S, g^t(s) \text{ } is \text{ } admissible \tag{3}$$

Let us prove that $\forall s \in S, g^{t+1}(s)$ is admissible as well. Let s be any state in S.

If $s \neq s_t$, then $g(s)$ is not modified during iteration t. Therefore, $g^{t+1}(s) = g^t(s)$, which is admissible by Equation 3. If $s = s_t$, then $g(s)$ is only modified by G-UPDATE (Step 3 of FALCONS). Now, if $s = s_{start}$, then $g^{t+1}(s) \overset{G-UPDATE}{=} g^t(s)$, which is admissible by Equation 3. Therefore, we need only consider the situation where $s = s_t \neq s_{start}$, for which it holds that:

$$g^{t+1}(s) \overset{G-UPDATE}{=} max \left\{ \begin{array}{l} g^t(s), \\ \min_{s'' \in pred(s)}(g^t(s'') + c(s'', s)), \\ \max_{s'' \in succ(s)}(g^t(s'') - c(s, s'')) \end{array} \right\}$$

We distinguish 3 cases, depending on which of the 3 arguments of max is the largest.

$$\text{Let } sp := \arg \min_{s'' \in pred(s)} (g^t(s'') + c(s'', s)). \tag{4}$$

$$\text{Let } ss := \arg \max_{s'' \in succ(s)} (g^t(s'') - c(s, s'')). \tag{5}$$

Case (i): $g^{t+1}(s) = g^t(s)$
Then, by Equation 3, $g^{t+1}(s)$ is admissible.

Case (ii):

$$g^{t+1}(s) = g^t(sp) + c(sp, s) \tag{6}$$

Proof by contradiction.

$$\text{Let } ssd := \arg\min_{s'' \in pred(s)} (sd(s'') + c(s'', s)). \tag{7}$$

$$\text{Thus, } sd(s) = sd(ssd) + c(ssd, s). \tag{8}$$

(Note that Equation 8 implies that $ssd \neq s$.) Now, assume $g^{t+1}(s) > sd(s)$.
This, combined with Equations 6 and 8, yields

$$g^t(sp) + c(sp, s) = g^{t+1}(s) > sd(s) = sd(ssd) + c(ssd, s). \tag{9}$$

But, since $g^t(ssd)$ is admissible by Equation 3, $sd(ssd) + c(ssd, s) \geq g^t(ssd) + c(ssd, s)$, which, combined with Equation 9, implies: $g^t(sp) + c(sp, s) > g^t(ssd) + c(ssd, s)$. The latter contradicts Equation 4. Therefore, $g^{t+1}(s) \leq sd(s)$, i.e. $g^{t+1}(s)$ is admissible.

Case (iii):

$$g^{t+1}(s) = g^t(ss) - c(s, ss) \tag{10}$$

(Note that Equation 10 implies that $ss \neq s$. Otherwise, $g(s) = g(ss)$ would strictly decrease between t and $t+1$, since $c(s, ss) \overset{A3}{>} 0$.) From $g^t(ss) \overset{\text{Equation 3}}{\leq} sd(ss)$ and $sd(ss) \overset{\text{def. of } sd}{\leq} sd(s) + c(s, ss)$, we obtain $g^t(ss) \leq sd(s) + c(s, ss)$, or equivalently $g^t(ss) - c(s, ss) \leq sd(s)$, which, combined with Equation 10, yields $g^{t+1}(s) \leq sd(s)$. Therefore, $g^{t+1}(s)$ is admissible.

In conclusion, $g^{t+1}(s)$ is admissible in all cases.

2. Proof by induction on t.
At $t = 0$, assumption A4 guarantees that $\forall s \in S$: $h^0(s)$ is admissible.
Assume that the induction hypothesis holds at the beginning of iteration t:

$$\forall s \in S\text{: } h^t(s) \textit{ is admissible} \tag{11}$$

Let us prove that $\forall s \in S$: $h^{t+1}(s)$ is admissible as well. Let s be any state in S.

If $s \neq s_t$, then $h(s)$ is not modified during iteration t. Therefore, $h^{t+1}(s) = h^t(s)$, which is admissible by Equation 11. If $s = s_t$, then $h(s)$ is only modified by H-UPDATE (Step 3 of FALCONS). Now, if $s = s_{goal}$, then $h^{t+1}(s) \overset{H-UPDATE}{=} h^t(s)$, which is admissible by Equation 11. Therefore, we need only consider the situation where $s = s_t \neq s_{goal}$, for which it holds that:

$$h^{t+1}(s) \overset{H-UPDATE}{=} max \left\{ \begin{array}{l} h^t(s), \\ \min_{s'' \in succ(s)}(c(s, s'') + h^t(s'')), \\ \max_{s'' \in pred(s)}(h^t(s'') - c(s'', s)) \end{array} \right\}$$

We distinguish 3 cases, depending on which of the 3 arguments of max is the largest.

$$\text{Let } ss \quad := \quad \arg\min_{s'' \in succ(s)} (c(s, s'') + h^t(s'')). \tag{12}$$

$$\text{Let } sp \quad := \quad \arg\max_{s'' \in pred(s)} (h^t(s'') - c(s'', s)). \tag{13}$$

Case (i): $h^{t+1}(s) = h^t(s)$
Then, by Equation 11, $h^{t+1}(s)$ is admissible.

Case (ii):

$$h^{t+1}(s) = c(s, ss) + h^t(ss) \tag{14}$$

Proof by contradiction.

$$\text{Let } sgd := \arg\min_{s'' \in succ(s)} (c(s, s'') + gd(s'')). \tag{15}$$

$$\text{Thus, } gd(s) = c(s, sgd) + gd(sgd). \tag{16}$$

(Note that Equation 16 implies that $sgd \neq s$.) Now, assume $h^{t+1}(s) > gd(s)$.
This, combined with Equations 14 and 16, yields

$$c(s, ss) + h^t(ss) = h^{t+1}(s) > gd(s) = c(s, sgd) + gd(sgd). \tag{17}$$

But, since $h^t(sgd)$ is admissible by Equation 11, $c(s, sgd) + gd(sgd) \geq c(s, sgd) + h^t(sgd)$, which, combined with Equation 17, implies $c(s, ss) + h^t(ss) > c(s, sgd) + h^t(sgd)$. The latter contradicts Equation 12. Therefore, $h^{t+1}(s) \leq gd(s)$, i.e. $h^{t+1}(s)$ is admissible.

Case (iii):

$$h^{t+1}(s) = h^t(sp) - c(sp, s) \tag{18}$$

(Note that Equation 18 implies that $sp \neq s$. Otherwise, $h(s) = h(sp)$ would strictly decrease between t and $t + 1$, since $c(sp, s) \overset{A3}{>} 0$.) From $h^t(sp) \overset{\text{Equation 11}}{\leq} gd(sp)$ and $gd(sp) \overset{\text{def. of } gd}{\leq} c(sp, s) + gd(s)$, we obtain $h^t(sp) \leq c(sp, s) + gd(s)$, or equivalently $h^t(sp) - c(sp, s) \leq gd(s)$, which, combined with Equation 18, yields $h^{t+1}(s) \leq gd(s)$. Therefore, $h^{t+1}(s)$ is admissible.

In conclusion, $h^{t+1}(s)$ is admissible in all cases. ■

Corollary 2

1. *Under assumptions A1-5, the g-values remain admissible during the execution of FALCONS without G-UPDATE.*
2. *Under assumptions A1-5, the h-values remain admissible during the execution of FALCONS without G-UPDATE.*

Proof:
1. Since G-UPDATE in Step 3 of FALCONS is the only step that modifies the g-values, FALCONS without G-UPDATE does not modify the g-values, and the g-values thus remain admissible.

2. Since G-UPDATE does not have any effect on the h-values, its absence in FALCONS does not make a difference in whether the h-values remain admissible. Therefore, this proof is the same as that for Lemma 2(2). ■

Lemma 3
Under assumptions A1-4, a trial of FALCONS could only run forever if, from some time on, it repeatedly moved along a finite cyclic path without modifying any of the g- and h-values in the cycle.

Proof: Consider the h-values. Lemma 1(2) guarantees that, on every transition, the h-value of the current state s can only increase or stay the same. In addition, Lemma 2(2) provides an upper bound on $h(s)$, namely $gd(s)$ (which is finite, by A2). This means that the maximum number of strict increases of $h(s)$ is finite. This reasoning holds for all states s in S. And since S is finite (A1), we infer that the maximum total number (over S) of strict increases of h-values by H-UPDATE is finite. The same reasoning applies to G-UPDATE for the g-values. In conclusion, there is a maximum, finite number of strict increases possible for both the g- and h-values. Therefore, if FALCONS never terminates, there must be a point in time, say T, after which no g- nor h-values are modified. Now, we prove that from some time T_1 on ($T_1 \geq T$), it must be the case that FALCONS repeatedly moves along a cycle. Let s^1 denote the first state to be visited twice after time T (s^1 must exist, by A1). Let T_1 (resp. T_2) denote the instant in time at which s^1 is reached for the first (resp. second) time after time T. By definition, $T_2 \geq T_1 \geq T$. Let C be the sequence of states (starting with s^1) travelled through in the time interval $[T_1, T_2)$. From time T_1 on, the cycle C is repeatedly followed by FALCONS. The reason for this is that no values in the state space changes after time T (and therefore after time T_1) and systematic tie-breaking (TB2) ensures that FALCONS will thereafter always choose the same successor at every decision point. ■

Corollary 3
Under assumptions A1-5, a trial of FALCONS without G-UPDATE could only run forever if, from some time on, it repeatedly moved along a finite cyclic path without modifying any of the g- and h-values in the cycle.

Proof: This proof is identical to that of Lemma 3, except that the finite number of strict increases of the g-values (namely zero) directly follows from the fact that FALCONS without G-UPDATE never modifies the g-values. ■

Lemma 4
Under assumptions A1-4, assume that FALCONS makes a transition from a state s_t to a state s_{t+1} without modifying the g- and h-values of s_t. Let $s'' := argmin_{s' \in succ(s_t)}(c(s_t, s') + h^t(s'))$. Then, s'' is such that:

1. $h^t(s'') \leq h^t(s_t) - c(s_t, s'')$,
2. $g^t(s'') \leq g^t(s_t) + c(s_t, s'')$,
3. $f^t(s'') \leq f^t(s_t)$,
4. $f^t(s_{t+1}) \leq f^t(s'')$, *and*
5. *if* $f^t(s_t) \leq f^t(s_{start})$, *then* $f^t(s_{t+1}) \leq f^t(s_{start})$.

Proof:

First, note that $s_t \neq s_{goal}$. Otherwise, FALCONS would stop in s_t.

Second, note that $s'' \neq s_t$. If that was not the case, it would hold that $h^{t+1}(s_t) \overset{H-UPDATE}{\geq} \min_{s' \in succ(s_t)}(c(s_t, s') + h^t(s')) \overset{Equation\ 19}{=} c(s_t, s_t) + h^t(s_t)$, which would imply that $h^{t+1}(s_t) - h^t(s_t) \geq c(s_t, s_t) \overset{A3}{>} 0$ and contradict our assumption that $h^{t+1}(s_t) = h^t(s_t)$. Thus, $s'' \neq s_t$.

$$\text{Let } s'' := argmin_{s' \in succ(s_t)}(c(s_t, s') + h^t(s')). \tag{19}$$

1. Proof by contradiction.
Assume $h^t(s_t) < c(s_t, s'') + h^t(s'')$. This, together with Equation 19, implies that $h^t(s_t) < \min_{s' \in succ(s_t)}(c(s_t, s') + h^t(s')) \overset{H-UPDATE}{\leq} h^{t+1}(s_t)$, which contradicts $h^{t+1}(s_t) = h^t(s_t)$. Therefore, s'' must satisfy $h^t(s_t) \geq c(s_t, s'') + h^t(s'')$, or equivalently $h^t(s'') \leq h^t(s_t) - c(s_t, s'')$.

2. Case (i): $s_t = s_{start}$

First, note that

$$\forall t \in \{1, 2, 3, \ldots\}\colon g^t(s_{start}) = 0 \tag{20}$$

follows from the fact that initially admissible g-values (A4) remain admissible (Lemma 2(1)).
$g^t(s'') \overset{Lemma\ 2(1)}{\leq} sd(s'')$ and $sd(s'') \overset{\text{def. of } sd}{\leq} sd(s_t) + c(s_t, s'') = c(s_t, s'')$ imply that $g^t(s'') \leq c(s_t, s'')$ or equivalently $g^t(s'') \leq 0 + c(s_t, s'') \overset{Equation\ 20}{=} g^t(s_{start}) + c(s_t, s'') = g^t(s_t) + c(s_t, s'')$.

Case (ii): $s_t \neq s_{start}$ (Proof by contradiction)

Assume $g^t(s_t) < g^t(s'') - c(s_t, s'')$.
This implies that $g^t(s_t) < \max_{s' \in succ(s_t)}(g^t(s') - c(s_t, s')) \overset{G-UPDATE}{\leq} g^{t+1}(s_t)$, which contradicts the assumption that $g^{t+1}(s_t) = g^t(s_t)$. Thus, $g^t(s_t) \geq g^t(s'') - c(s_t, s'')$ or equivalently $g^t(s'') \leq g^t(s_t) + c(s_t, s'')$.

3. From Results 1 and 2 above, we have $g^t(s'') + h^t(s'') \leq g^t(s_t) + c(s_t, s'') + h^t(s_t) - c(s_t, s'')$ or equivalently $g^t(s'') + h^t(s'') \leq g^t(s_t) + h^t(s_t)$. Thus, $max(h^t(s_{start}), g^t(s'') + h^t(s'')) \leq max(h^t(s_{start}), g^t(s_t) + h^t(s_t))$ which, by F-CALC, is equivalent to $f^t(s'') \leq f^t(s_t)$.

4. Since FALCONS chooses s_{t+1} as the next state, it must hold that $f^t(s_{t+1}) \leq f^t(s'')$.

5. Assume $f^t(s_t) \leq f^t(s_{start})$. This, together with Result 3 above, implies that

$$f^t(s'') \leq f^t(s_{start}). \tag{21}$$

Since FALCONS chooses to move to s_{t+1}, it must be the case that $f^t(s_{t+1}) \leq f^t(s'')$, which, together with Equation 21, yields $f^t(s_{t+1}) \leq f^t(s_{start})$. ■

Corollary 4
Under assumptions A1-5, assume that FALCONS without G-UPDATE makes a transition from a state s_t to a state s_{t+1} without modifying the g- and h-values of s_t. Let $s'' := argmin_{s' \in succ(s_t)}(c(s_t, s') + h^t(s'))$. Then, s'' is such that:

1. $h^t(s'') \leq h^t(s_t) - c(s_t, s'')$,
2. $g^t(s'') \leq g^t(s_t) + c(s_t, s'')$,
3. $f^t(s'') \leq f^t(s_t)$,
4. $f^t(s_{t+1}) \leq f^t(s'')$, *and*
5. *if* $f^t(s_t) \leq f^t(s_{start})$, *then* $f^t(s_{t+1}) \leq f^t(s_{start})$.

Proof:
For the same reasons as in the proof for Lemma 4, $s_t \neq s_{goal}$ and $s'' \neq s_t$. In addition, the g-values are never modified.

1. This proof is the same as that for Lemma 4(1).

2. Case (i): $s_t = s_{start}$
This proof is the same as that for Lemma 4(2), except that Equation 20 is now true because of A4 and the absence of G-UPDATE, and that we use Corollary 2(1) instead of Lemma 2(1).

Case (ii): $s_t \neq s_{start}$
$g^t(s'') \leq g^t(s_t) + c(s_t, s'')$ directly follows from A5 and the definition of the consistency of the g-values.

3. This proof is the same as that for Lemma 4(3) above.

4. This proof is the same as that for Lemma 4(4) above.

5. This proof is the same as that for Lemma 4(5) above. ■

Lemma 5
Under assumptions A1-4, assume that FALCONS follows a path P starting in any state s^1 without modifying the g- and h-values of any state on P. If $f(s^1) \leq f(s_{start})$, then for all states s on P,

$$f(s) \leq f(s_{start}). \tag{22}$$

Proof: Proof by induction on the distance of s from s^1 on P.
If $s = s^1$, then $f(s) = f(s^1) \leq f(s_{start})$. So, Equation 22 trivially holds for s^1.
Assume that s is any state on P but the last one. Then, s has a successor s' on P. Lemma 4(5) directly allows us to infer that, if Equation 22 holds for s, then it also holds for s'. ■

Corollary 5
Under assumptions A1-5, assume that FALCONS without G-UPDATE follows a path P starting in any state s^1 without modifying the g- and h-values of any state on P. If $f(s^1) \leq f(s_{start})$, then for all states s on P, s on P,

$$f(s) \leq f(s_{start}). \tag{23}$$

Proof: The proof is the same as that for Lemma 5, except that it uses Corollary 4(5) instead of Lemma 4(5). ■

Lemma 6
Under assumptions A1-4, at all times t during the execution of FALCONS, $f^t(s_{start}) = h^t(s_{start})$.

Proof:
$f^t(s_{start}) \overset{F-CALC}{=} max(g^t(s_{start}) + h^t(s_{start}), h^t(s_{start})) \overset{A4+Lemma\ 2(1)}{=} max(0 + h^t(s_{start}), h^t(s_{start})) = h^t(s_{start})$. ■

Corollary 6
Under assumptions A1-5, at all times t during the execution of FALCONS without G-UPDATE, $f^t(s_{start}) = h^t(s_{start})$.

Proof:
The proof is the same as that for Lemma 6, except that it uses Corollary 2(1) instead of Lemma 2(1). ■

Theorem 1 (Termination 1)
Under assumptions A1-4, each trial of FALCONS is guaranteed to terminate.

Proof: Proof by contradiction.
If FALCONS cycles forever then there exists a finite cyclic path P along which the g- and h-values do not change from some time T on (Lemma 3). In the following, we can drop the superscripts on the h- and f-values since they do not change after time T.

We distinguish two cases. Either all states on P have f-values smaller than or equal to $f(s_{start})$, or all states on P have f values greater than $f(s_{start})$. These are the only two possible cases. Indeed, if there is at least one state s^1 on P such that $f(s^1) \leq f(s_{start})$, then all states following s^1 on P will also have an f-value smaller than or equal to $f(s_{start})$ (by Lemma 5). But, since P is cyclic, every state s on P follows s^1 and therefore satisfies $f(s) \leq (s_{start})$.

Case (i): For all states s_t on the cycle, $f(s_t) > f(s_{start})$.
In this case, it must hold that for all successors s' of all states in the cycle, $f(s') > f(s_{start})$. Otherwise, FALCONS would choose as next state a successor with $f(s') \leq f(s_{start})$ and thus leave the cycle.
Let s_t be any state on this cycle, s_{t+1} be the successor of s on the cycle, and $s'' := argmin_{s' \in succ(s_t)}(c(s_t, s') + h(s'))$.
By Lemma 4(3&4), $f(s_{t+1}) \leq f(s_t)$, i.e. the f-values cannot increase along a transition. Therefore they cannot decrease either because otherwise they would have to increase again before the end of the cycle. So the f-values of all states on the cycle are the same and in particular $f(s_{t+1}) = f(s_t)$ which, combined with Lemma 4(3&4) yields $f(s_{t+1}) = f(s'') = f(s_t)$. Since FALCONS chooses s_{t+1} as the next state, $c(s_t, s_{t+1}) + h(s_{t+1}) \overset{TB}{\leq} c(s_t, s'') + h(s'')$. By definition of s'' and H-UPDATE (since $h(s_t)$ does not change), $c(s_t, s'') + h(s'') \leq h(s_t)$ Combining the two previous inequalities yields $c(s_t, s_{t+1}) + h(s_{t+1}) \leq h(s_t)$. Since $c(s_t, s_{t+1}) \overset{A3}{>} 0$, it follows that $h(s_{t+1}) < h(s_t)$. This means that the h-value strictly decreases along this and therefore all transitions on the cycle, which is impossible.

Case (ii): For all states s_t on the cycle, $f(s_t) \leq f(s_{start})$.
Let s_t be any state on the cycle. Let s_{t+1} be the successor of s on the cycle and $s'' := argmin_{s' \in succ(s_t)}(c(s_t, s') + h(s'))$.
Now, $f(s'') \overset{Lemma\ 4(3)}{\leq} f(s_t)$ and $f(s_t) \leq f(s_{start})$ imply that $f(s'') \leq f(s_{start}) \overset{Lemma\ 6}{=} h(s_{start})$, which, combined with $f(s'') \overset{F-CALC}{\geq} h(s_{start})$, yields

$$f(s'') = h(s_{start}). \tag{24}$$

Furthermore, $f(s_{t+1}) \overset{Lemma\ 4(5)}{\leq} f(s_{start}) \overset{Lemma\ 6}{=} h(s_{start})$ implies, together with $f(s_{t+1}) \overset{F-CALC}{\geq} h(s_{start})$, that $f(s_{t+1}) = h(s_{start})$. Combining this equation with Equation 24 yields $f(s'') = f(s_{t+1})$. Now, $c(s_t, s_{t+1}) + h(s_{t+1}) \overset{TB}{\leq} c(s_t, s'') + h(s'')$. In addition, since $h(s_t)$ does not change after the update, we know that $c(s_t, s'') + h(s'') \leq h(s_t)$. Chaining the two together, we get $c(s_t, s_{t+1}) + h(s_{t+1}) \leq h(s_t)$ or equivalently $h(s_{t+1}) \leq h(s_t) - c(s_t, s_{t+1}) < h(s_t)$, since $c(s_t, s_{t+1}) \overset{A3}{>} 0$. This means that the h-value strictly decreases along this and therefore all transitions in the cycle, which is impossible. ■

Theorem 2 (Termination 2)
Under assumptions A1-5, each trial of FALCONS without G-UPDATE is guaranteed to terminate.

Proof: The proof for this theorem is the same as that for Theorem 1 except that it uses the corollaries instead of the lemmata with the corresponding numbers. ■

Lemma 7 (Convergence)
Under assumptions A1-4, assume FALCONS is reset to s_{start} at the end of each trial and the g- and h-values are kept from each trial to the next. Then, from some time on, FALCONS will always follow the same path.

Proof:
Theorem 1 above has established that each trial of FALCONS will always terminate. We now assume that FALCONS is reset into s_{start} at the end of each trial. We can follow a reasoning similar to that used in the proof of Lemma 3 to establish that from some time T on, no g- and h-value will change any longer. This is because these values can only increase or remain unchanged (Lemma 1) and remain admissible (Lemma 2). Therefore, the g- and h-values are bounded from above by finite values (by A2) and cannot increase forever. Now, let t_1 denote the first trial that starts after time T and let P denote the path followed during t_1. Since no g- nor h-value changes during t_1, the next trial, say t_2, will start with the same heuristic values. And since remaining ties are broken systematically (TB2), FALCONS, starting at the same state s_{start}, will necessarily follow the same path P during t_2. The same reasoning holds for all subsequent trials. Therefore, from trial t_1 on, FALCONS will always follow the same path P. It has therefore converged to P. ■

Corollary 7
Under assumptions A1-5, assume FALCONS without G-UPDATE is reset to s_{start} at the end of each trial and the g- and h-values are kept from each trial to the next. Then, from some time on, FALCONS without G-UPDATE will always follow the same path.

Proof:
The proof is identical to that for Lemma 7 except that it uses Theorem 2 instead of Theorem 1 and the corollaries corresponding to the lemmata. ■

Theorem 3 (Convergence to a shortest path 1)
Under assumptions A1-4, FALCONS converges to a shortest path.

Proof:
(In this proof, the time superscript of the f- and h-values are omitted for ease of reading.) Assume that FALCONS has converged to a path P from s_{start} to s_{goal} (Lemma 7).
Since the first state in P is s_{start} and its f-value is trivially less than or equal to $f(s_{start})$, we can use Lemma 5 to infer that, for all states s_t on P, $f(s_t) \leq f(s_{start})$. Let us consider any state s_t on P, s_{t+1} its successor on P, and let $s'' := argmin_{s' \in succ(s_t)}(c(s_t, s') + h(s'))$.
By combining Lemma 4(3) with $f(s_t) \leq f(s_{start})$, we get $f(s'') \leq f(s_t) \leq f(s_{start}) \overset{Lemma\ 6}{=} h(s_{start})$, which, combined with $f(s'') \overset{F-CALC}{\geq} h(s_{start})$ yields $f(s'') = h(s_{start})$. Similarly, $f(s_{t+1}) \overset{Lemma\ 4(5)}{\leq} f(s_{start}) \overset{Lemma\ 6}{=} h(s_{start})$ and $f(s_{t+1}) \overset{F-CALC}{\geq} h(s_{start})$ yield $f(s_{t+1}) = h(s_{start})$. Thus, $f(s'') = f(s_{t+1})$. Since the chosen successor is s_{t+1}, it must be the case (by TB) that $h(s_{t+1}) + c(s_t, s_{t+1}) \leq h(s'') + c(s_t, s'')$.

According to Lemma 4(1), $h(s'') + c(s_t, s'') \leq h(s_t)$. Combining the last two inequalities, we get $h(s_{t+1}) + c(s_t, s_{t+1}) \leq h(s_t)$, or equivalently

$$h(s_t) - h(s_{t+1}) \geq c(s_t, s_{t+1}). \tag{25}$$

Adding up the instances of Equation 25 for each transition on P yields $h(s_{start}) - h(s_{goal}) \geq cost_P(s_{start}, s_{goal})$, where $cost_P(s_{start}, s_{goal})$ denotes the total cost of path P. Since $h(s_{goal}) \overset{A4+Lemma\ 2(2)}{=} 0$, we infer $h(s_{start}) \geq cost_P(s_{start}, s_{goal})$. Now, the definition of the goal distance implies that $cost_P(s_{start}, s_{goal}) \geq gd(s_{start})$. Finally, admissibility of h means that $gd(s_{start}) \geq h(s_{start})$. Chaining the last three inequalities, we get $gd(s_{start}) \geq h(s_{start}) \geq cost_P(s_{start}, s_{goal}) \geq gd(s_{start})$ and conclude that $cost_P(s_{start}, s_{goal}) = gd(s_{start})$. Therefore, P is a minimum-cost path from s_{start} to s_{goal}, which means that FALCONS has converged to a shortest path. ■

Theorem 4 (Convergence to a shortest path 2)
Under assumptions A1-5, FALCONS without G-UPDATE converges to a shortest path.

Proof: The proof is the same as that for Theorem 3 except that the corollaries are used instead of the corresponding lemmata. ■

References

Furcy, D., and Koenig, S. 2000. Speeding up the convergence of real-time search. In *Proceedings of the National Conference on Artificial Intelligence*.

Holte, R.; Drummond, C.; Perez, M.; Zimmer, R.; and MacDonald, A. 1994. Searching with abstractions: A unifying framework and new high-performance algorithm. In *Proceedings of the Canadian Conference on Artificial Intelligence*, 263–270.

Holte, R.; Perez, M.; Zimmer, R.; and MacDonald, A. 1996. Hierarchical A*: Searching abstraction hierarchies efficiently. In *Proceedings of the National Conference on Artificial Intelligence*, 530–535.

Ishida, T. 1997. *Real-Time Search for Learning Autonomous Agents*. Kluwer Academic Publishers.

Korf, R. 1980. Towards a model of representation changes. *Artificial Intelligence* 14:41–78.

Korf, R. 1990. Real-time heuristic search. *Artificial Intelligence* 42(2-3):189–211.

Nilsson, N. 1971. *Problem-Solving Methods in Artificial Intelligence*. McGraw-Hill.