

# PARA\*: Parallel Anytime Repairing A\*

AAAI 2015 Submission X

## Abstract

PARA\* is an anytime parallel heuristic search algorithm based on ARA\* and PA\*SE, which are in turn based on A\*.

## Fancy Stuff

---

### Algorithm 1 bound(s)

---

```

 $g_{front} := \infty$ 
 $s' := \text{first node in } OPEN \cup BE$ 
 $g_{back} := g(s) + f(s') - f(s) + (2\epsilon - w - 1)c_l$ 
while  $g_{back} < g(s) \leq g_{front}$  do
     $g_{front} := \min(g_{front}, g_p(s') + \epsilon h(s', s))$ 
     $s' := \text{node following } s' \text{ in } OPEN \cup BE$ 
     $g_{back} := g(s) + f(s') - f(s) + (2\epsilon - w - 1)c_l$ 
end while
return  $\min(g_{front}, g_{back})$ 

```

---

Keys are always computed by  $f(s) = g(s) + wh(s, s_{goal})$ , and we assume all edge costs are bounded below by  $c_l$ .  $h$  must be consistent:  $h(s, s') \leq c(s, s')$  and  $h(s, s') \leq h(s, s'') + h(s'', s')$  for all  $s, s', s''$ . For most applications, we recommend using  $w = \epsilon$ . However, our analysis will show that using small  $w$  yields strong parallelism guarantees. All operations on the data structures *OPEN*, *BE*, *CLOSED*, *FROZEN* are assumed to be atomic, i.e. they are implicitly preceded and succeeded by synchronous locks and unlocks to the data structure, respectively.  $\epsilon$  decreases between iterations of the main() loop.  $v(s)$  is included to aid the analysis but is never used in the algorithm. In main(), the loops involving all  $s \notin OPEN$  are computed lazily when those states are first encountered.

**Lemma 1.** *At all times, the following invariants hold:*

- $BE \subset CLOSED$  and  $FROZEN \subset CLOSED$
- $OPEN \cap CLOSED = \emptyset$
- $s \in OPEN \cup BE \cup FROZEN \Rightarrow g(s) < v(s)$
- $s \notin OPEN \cup BE \cup FROZEN \Rightarrow g(s) = v(s)$
- $g(bp(s)) + c(bp(s), s) \leq g(s) \leq \min_{s'} \{v(s') + c(s', s)\}$

Copyright © 2014, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

---

### Algorithm 2 expand(s)

---

```

for all  $s' \in \text{successors}(s)$  do
    LOCK  $s'$ 
    if  $s'$  was not yet seen in this main() iteration then
         $g_p(s') := \infty$ 
        if  $s'$  has not been generated yet then
             $g(s') := v(s') := \infty$ 
        end if
    end if
    if  $g_p(s') = \min(g_p(s'), g_{bound} + \epsilon c(s, s'))$ 
    if  $g(s') > g(s) + c(s, s')$  then
         $g(s') = g(s) + c(s, s')$ 
         $bp(s') = s$ 
        if  $s' \in CLOSED$  then
            insert  $s'$  in FROZEN
        else
            insert/update  $s'$  in OPEN with key  $f(s')$ 
        end if
    end if
    UNLOCK  $s'$ 
end for

```

---



---

### Algorithm 3 PARA\*

---

```

while  $g(s_{goal}) > bound(s_{goal})$  do
    among  $s \in OPEN$  such that  $g(s) \leq bound(s)$ , re-
    move one with the smallest  $f(s)$  and LOCK  $s$ 
    if such an  $s$  does not exist then
        wait until OPEN or BE change
        continue
    end if
     $g_{bound} := bound(s)$ 
     $v_{expand} := g(s)$ 
    insert  $s$  into CLOSED
    insert  $s$  into BE with key  $f(s)$ 
    UNLOCK  $s$ 
    expand(s)
     $v(s) := v_{expand}$ 
    remove  $s$  from BE
end while

```

---

**Algorithm 4** main()

---

```

OPEN := BE := ∅
FROZEN := {s_start}
g(s_start) := 0
repeat
  choose  $\epsilon \in [1, \infty]$  and  $w \in [0, \epsilon]$ 
  OPEN := OPEN  $\cup$  FROZEN with keys  $f(s)$ 
  CLOSED := FROZEN := ∅
  for all  $s \in OPEN$  do
     $g_p(s) := \epsilon g(s)$ 
  end for
  run PARA* on multiple threads in parallel
until path is good enough or planning time runs out

```

---

- Following  $bp(\cdot)$  from  $s$  yields a path costing at most  $g(s)$
- $s \neq s_{start} \Rightarrow g(s) + (\epsilon - 1)c_l \leq g_p(s)$
- $s \in OPEN \cup CLOSED \Rightarrow g_p(s) \leq \epsilon g(s)$

*Proof.* Induction on time.  $\square$

**Lemma 2.** At all times, for all states  $s, s' \neq s_{start}$ :

$$g(s) + f(s') - f(s) + t \leq g_p(s') + \epsilon h(s', s).$$

*Proof.*

$$\begin{aligned}
& g(s) + f(s') - f(s) + (2\epsilon - w - 1)c_l \\
&= g(s') + w(h(s', s_{goal}) - h(s, s_{goal})) + (2\epsilon - w - 1)c_l \\
&\leq g(s') + wh(s', s) + (2\epsilon - w - 1)c_l \\
&\leq g(s') + \epsilon h(s', s) + (w - \epsilon)c_l + (2\epsilon - w - 1)c_l \\
&= g(s') + (\epsilon - 1)c_l + \epsilon h(s', s) \\
&\leq g_p(s') + \epsilon h(s', s)
\end{aligned}$$

$\square$

**Lemma 3.**  $bound(s) \leq \min_{s' \in OPEN \cup BE} g_p(s') + \epsilon h(s', s)$ . Furthermore,  $g(s) \leq bound(s)$  iff  $g(s) \leq \min_{s' \in OPEN \cup BE} g_p(s') + \epsilon h(s', s)$ .

*Proof.* By construction,  $bound(s)$  is bounded above by  $g_p(s') + \epsilon h(s', s)$  for states  $s'$  which are checked in the loop. As for the remaining states  $s' \in OPEN \cup BE$ , the algorithm ensures that  $bound(s) \leq g(s) + f(s') - f(s) + (2\epsilon - w - 1)c_l$  for these by using a minimum representative. By Lemma 2, it follows that

$$bound(s) \leq \min_{s' \in OPEN \cup BE} g_p(s') + \epsilon h(s', s).$$

For the second part, note that the loop in  $bound(s)$  terminates under only two conditions. Either  $g(s) > g_{front}$ , in which case we have  $g(s) > g_p(s') + \epsilon h(s', s) \geq bound(s)$  for the  $s'$  which began the final iteration; or  $g(s) \leq g_{back}$ , in which case  $g(s) \leq bound(s)$  iff  $g(s) \leq g_{front}$  iff  $g(s) \leq g_p(s') + \epsilon h(s', s)$  for all  $s' \in OPEN \cup BE$ .  $\square$

**Theorem 1.** For all states  $s$ ,  $bound(s) \leq \epsilon g^*(s)$ . Hence, for all  $s \in CLOSED$ ,  $g(s) \leq v(s) \leq \epsilon g^*(s)$ .

*Proof.* We proceed by induction on the order in which states are expanded.

Let  $\pi = \langle s_0, s_1, \dots, s_N \rangle$  be a minimum-cost path from  $s_0 = s_{start}$  to  $s_N = s$ . There are three cases to consider, our first branch depending on whether there exists  $s_i \in \pi$  such that  $s_i \in CLOSED \setminus BE$ .

If so, choose the maximum such  $i$ . Since  $s_i$  was expanded and yet  $s_{i+1} \notin CLOSED \setminus BE$ , it follows that  $s_{i+1} \in OPEN \cup BE$  and

$$g_p(s_{i+1}) \leq \epsilon(g^*(s_i) + c(s_i, s_{i+1})) = \epsilon g^*(s_{i+1})$$

One the other hand, suppose there is no  $s_i \in CLOSED \setminus BE$ . Suppose also that every  $s_i \in OPEN \cup BE$  was opened by an expansion. Then the entire path  $\pi$  was consistent when this main() loop iteration started; therefore,  $g_p(s) \leq \epsilon g(s) = \epsilon g^*(s)$  and we are done.

In the final case, choose the minimum  $i$  such that  $s_i \in OPEN \cup BE$  and  $s_i$  was opened by “thawing” the FROZEN list. Then  $g(s_i) = g^*(s_i)$ . Therefore,

$$g_p(s_i) \leq \epsilon g(s_i) = \epsilon g^*(s_i)$$

In the first and third cases, we found an  $s' \in OPEN \cup BE$  such that

$$g_p(s') + \epsilon h(s', s) \leq \epsilon(g^*(s') + c^*(s', s)) = \epsilon g^*(s).$$

Therefore, by Lemma 3,

$$bound(s) \leq \min_{s' \in OPEN \cup BE} g_p(s') + \epsilon h(s', s) \leq \epsilon g^*(s).$$

$\square$

**Corollary 1.** At the end of a main() loop iteration, the path obtained by following the back-pointers  $bp(\cdot)$  from  $s_{goal}$  to  $s_{start}$  is  $\epsilon$ -suboptimal.

*Proof.* The termination condition of PARA\* implies  $g(s_{goal}) \leq bound(s_{goal})$ . By construction, the path given by following back-pointers costs at most  $g(s_{goal})$ . The claim now follows from Theorem 1.  $\square$

## Performance Guarantees

Consider a simplified version of PARA\* which ignores the loop in  $bound(s)$ : we call it blind PARA\*. In this case, no  $g_p$  values need be computed nor stored, and  $bound(s)$  is simply  $g(s) + f_{min} - f(s) + (2\epsilon - w - 1)c_l$  where  $f_{min}$  is the minimum  $f$ -value in  $OPEN \cup BE$ . Blind PARA\* can only expand states which would be proved safe with zero iterations of the  $bound(s)$  loop in ordinary PARA\*. Thus, all of the performance guarantees we prove for blind PARA\* also hold for PARA\*.

**Theorem 2.** If  $w \leq 1$ , the parallel depth of blind PARA\* is bounded above by

$$\min \left( \frac{\epsilon g^*(s_{goal})}{(1-w)c_l}, \frac{(\epsilon g^*(s_{goal}))^2}{(4\epsilon - 2w - 2)c_l^2} \right).$$

*Proof.* We prove the two bounds separately. For the first, note that if the lowest  $f$ -value is  $f_{min}$ , every state with  $f$ -value up to  $f_{min} + (2\epsilon - w - 1)c_l$  can simultaneously be expanded. Since  $h$  is consistent, the successors'  $f$ -values is at least  $f_{min} + (1 - w)c_l$ . Therefore, the depth is at most

$$\frac{\epsilon g^*(s_{goal})}{(1 - w)c_l}$$

For the other bound, notice that since  $f$ -values never decrease along paths, once the minimum  $f$ -value in *OPEN* surpasses  $f_{min}$ , from then on all nodes with  $f$ -value up to  $f_{min} + (2\epsilon - w - 1)c_l$  are always safe to expand. And during each iteration of the simultaneous expansions, the  $g$ -value of all such nodes increases by at least  $c_l$ . Since  $g$  cannot exceed  $f$ , this continues for at most  $(f_{min} + (2\epsilon - w - 1)c_l)/c_l = f_{min}/c_l + 2\epsilon - w - 1$  iterations, after which every node in *OPEN* has  $f$ -value  $\geq f_{min} + (2\epsilon - w - 1)c_l$ . Continuing this process until  $f_{min}$  exceeds  $\epsilon g^*(s_{goal})$ , a bound on the total iteration count is:

$$2\epsilon - w - 1 + 2(2\epsilon - w - 1) + 3(2\epsilon - w - 1) + \dots + \epsilon g^*(s_{goal})/c_l \cong (\epsilon g^*(s_{goal})/c_l)^2 / (4\epsilon - 2w - 2). \quad \square$$

### Edgewise Supoptimality

Let  $k(s)$  be the least number of edges used in a minimum-cost path to  $s$  and fix  $\delta > 0$ . If  $g_{front}$  and  $g_{back}$  are each increased by  $2\delta$ , then by similar arguments to the proofs earlier in the paper, we find that, upon expanding  $s$ ,  $g(s) \leq \epsilon g^*(s) + \delta k(s)$ .

Here's an extension inspired by (Klein and Subramanian 1997): suppose the mean edge cost  $c_m$  along the optimal path is known to be much greater than the lower bound  $c_l$ . In such a case, the bound in Theorem 2 scales poorly. To remedy the situation, we "grow" the small edges, effectively running *PARA\** with  $c'_l = c_l + \delta$  and  $c'(s, s') = \max(c(s, s'), c'_l)$ .

**Theorem 3.** *If the mean cost of the edges along the minimum-cost path to  $s$  is at least  $c_m$ , then upon expansion,  $g(s) \leq \epsilon(1 + \delta/c_m)g^*(s)$ . Therefore, to get the same optimality factor as  $\epsilon$ , we can set  $\delta = (\epsilon - 1)c_m$ .*

*Proof.* We assumed  $c_m \leq g^*(s)/k(s)$ , so  $k(s) \leq g^*(s)/c_m$ . It follows from Lemma 1 that  $g'(s) \leq \epsilon g'^*(s) \leq \epsilon(g^*(s) + \delta k(s)) \leq \epsilon(1 + \delta/c_m)g^*(s)$ .  $\square$

**Corollary 2.** *If  $w \leq 1$ , the parallel depth of blind *PARA\** can be improved to*

$$\frac{\epsilon g^*(s_{goal})}{(1 - w)(c_l + (\epsilon - 1)c_m)}.$$

### References

Klein, P. N., and Subramanian, S. 1997. A randomized parallel algorithm for single-source shortest paths. *Journal of Algorithms* 25(2):205–220.