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 Auxiliary Functions

FUNCTION $f(s)$
return $g(s) + wh(s, s_{goal})$
FUNCTION $g_{back}(s', s)$
if $w \leq \epsilon$ **then**
  return $g(s) + f(s') - f(s) + (2\epsilon - w - 1)c_l$
else
  return $\frac{\epsilon}{w}(g(s) + f(s') - f(s)) + (\epsilon - 1)c_l$
end if
FUNCTION $bound(s)$
$g_{front} := \infty$
s' := first node in $OPEN \cup BE$
while $g_{back}(s', s) < g(s) \leq g_{front}$ **do**
  $g_{front} := \min(g_{front},\ g_p(s') + \epsilon h(s', s))$
  s' := node following s' in $OPEN \cup BE$
end while
return $\min(g_{front},\ g_{back}(s', s))$

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. $v(s)$ is included to aid the analysis but is never used in the algorithm.

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

- $OPEN \cap CLOSED = \emptyset$
- $BE \cup FROZEN \subset CLOSED$

Algorithm 2 expand(s)

for all $s' \in successors(s)$ **do**
  LOCK s'
  if s' was not yet seen in this main() iteration **then**
    if s' has not been generated yet **then**
      $g(s') := v(s') := \infty$
    end if
    $g_p(s') := g(s') + 2(\epsilon - 1)c_l$
  end 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 $g(s) \leq bound(s)$, remove 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()

$OPEN := BE := \emptyset$
$FROZEN := \{s_{start}\}$
$g(s_{start}) := 0$
repeat
  choose $\epsilon \in [1, \infty]$ and $w \in [0, \infty]$
  $OPEN := OPEN \cup FROZEN$ with keys $f(s)$
  $CLOSED := FROZEN := \emptyset$
  for all $s \in OPEN$ **do**
    $g_p(s) := g(s) + 2(\epsilon - 1)c_l$
  end for
  run PARA* on multiple threads in parallel
until path is good enough or planning time runs out

- $s \in OPEN \cup BE \cup FROZEN \Rightarrow g(s) < v(s)$
- $s \notin OPEN \cup BE \cup FROZEN \Rightarrow g(s) = v(s)$
- $g(bp(s)) + c(bp(s), s) \leq g(s) \leq min_{s'}\{v(s') + c(s', s)\}$
- *Following* $bp(\cdot)$ *from* s *yields a path costing at most* $g(s)$
- $g(s) + (\epsilon - 1)c_l \leq g_p(s)$
- $s \in OPEN \cup CLOSED$ *iff we had* $g(s) < v(s)$ *sometime during the current main() loop iteration*

Proof. Induction on time. □

Lemma 2. *At all times, for all states* s, s'*:*

$$g_{back}(s', s) \leq g_p(s') + \epsilon h(s', s).$$

Proof. If $w \leq \epsilon$, then

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

On the other hand, if $w > \epsilon$, then

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

□

Lemma 3. *For all states* s, $bound(s) \leq \min_{s' \in OPEN \cup BE} g_p(s') + \epsilon h(s', s)$*. 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 $bound(s) \leq g_{back}(s', s)$ for these by using a minimum representative. By Lemma 2, it follows that

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

To prove the second claim, 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 $s' \in OPEN \cup BE$. □

Theorem 1. *For all* $s \in OPEN \cup BE$*, if* s *is not* s_{start} *or one of its optimal-path successors, then* $bound(s) \leq \epsilon g^*(s)$*. 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 $\pi = \langle s_0, s_1, \ldots, s_N \rangle$ be a minimum-cost path from $s_0 = s_{start}$ to $s_N = s \in OPEN \cup BE$. Choose the minimum i such that $s_i \in OPEN \cup BE$. There are two cases to consider, depending on whether $s_{i-1} \in CLOSED$.

If so, then $expand(s_{i-1})$ has assigned to $g_p(s_i)$. Hence by the induction hypothesis,

$$\begin{aligned}
g_p(s_i) &\leq v(s_{i-1}) + \epsilon c(s_{i-1}, s_i) \\
&\leq \epsilon g^*(s_{i-1}) + \epsilon c(s_{i-1}, s_i) \\
&= \epsilon g^*(s_i)
\end{aligned}$$

On the other hand, suppose $s_{i-1} \notin CLOSED$. Choose the maximum $j < i$ such that $s_j \in CLOSED$, or $j = 0$ if there is no such j. Then $j \leq i - 2$ and, by the induction hypothesis, $g(s_j) \leq \epsilon g^*(s_j)$. Furthermore, $g(s_k) = v(s_k)$ for all $j < k < i$. Let $g_{old}(s_i)$ denote the value of $g(s_i)$ at the start of the current main() loop iteration. Then,

$$\begin{aligned}
g_p(s_i) &\leq g_{old}(s_i) + 2(\epsilon - 1)c_l \\
&\leq v(s_j) + c^*(s_j, s_i) + 2(\epsilon - 1)c_l \\
&\leq \epsilon g^*(s_j) + c^*(s_j, s_i) + 2(\epsilon - 1)c_l \\
&= \epsilon(g^*(s_j) + c^*(s_j, s_i)) + (\epsilon - 1)(2c_l - c^*(s_j, s_i)) \\
&\leq \epsilon g^*(s_i)
\end{aligned}$$

In both cases, we found that

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

Therefore, by Lemma 3,

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

□

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* ϵ*-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. □

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:

$2\epsilon - w - 1 + 2(2\epsilon - w - 1) + 3(2\epsilon - w - 1) + ... + \epsilon g^*(s_{goal})/c_l \approxeq (\epsilon g^*(s_{goal})/c_l)^2/(4\epsilon - 2w - 2)$. □

Edgewise Supobtimality

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 2δ, 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 ϵ, 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 $g'(s) \leq \epsilon g'^*(s) \leq \epsilon(g^*(s) + \delta k(s)) \leq \epsilon(1 + \delta/c_m)g^*(s)$. □

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.