PAPA*: Path-Aware Parallel A*

AAAI 2015 Submission X

Abstract

PAPA* 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)

> $g_{front} := \infty$
> $s' :=$ 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**
> $g_{front} := \min(g_{front},\ g_p(s') + \epsilon h(s', s))$
> $s' :=$ node following s' 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. ϵ decreases between iterations of the main() loop.

Lemma 1. *At all times, for all states $s, s' \neq s_{start}$:*

$$g(s) + f(s') - f(s) + (2\epsilon - w - 1)c_l \leq g_p(s') + \epsilon h(s', s).$$

Proof.

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

Algorithm 2 PAPA*

> **while** $g(s_{goal}) > bound(s_{goal})$ **do**
>   remove an s from $OPEN$ that has the smallest $f(s)$ among all states in $OPEN$ with $g(s) \leq bound(s)$ and let $g_{bound} := bound(s)$
>   **if** such an s does not exist **then**
>   wait until $OPEN$ or BE change
>   continue
>   **end if**
>   insert s into BE
>   insert s into $CLOSED$
> $S := getSuccessors(s)$
>   **for all** $s' \in S$ **do**
>   LOCK s'
>   **if** s' has not been generated yet **then**
> $g(s') := g_p(s') := \infty$
>   **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**
>   remove s from BE
> **end while**

Algorithm 3 main()

```
g(s_start) := 0
g(s_goal) := g_p(s_goal) := ∞
OPEN := BE := ∅
FROZEN := {s_start}
repeat
   choose ϵ ∈ [1, ∞] and w ∈ [0, ϵ]
   OPEN := OPEN ∪ FROZEN with keys f(s)
   CLOSED := FROZEN := ∅
   for all s ∈ OPEN do
      g_p(s) := ϵg(s)
   end for
   run PAPA* on multiple threads in parallel
until path is good enough or planning time runs out
```

□

Lemma 2. *$bound(s) \leq \min_{s' \in OPEN \cup BE} g_p(s') + \epsilon h(s', s)$. 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 1, 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 $s' \in OPEN \cup BE$. □

Lemma 3. *For every optimal path from s_{start} to some $s \in OPEN$, every state s' up to and including the first one in $OPEN \cup BE \cup FROZEN$ has $g(s') = g^*(s')$.*

Proof. We proceed by induction on time: noting the lemma holds right after s_{start} is added to the open list, we show that it can never become false. Suppose for contradiction that it becomes false at some point. Since $g(s')$ never changes after achieving $g^*(s')$, it must be the case that the first state s' along some optimal path to lie in $OPEN \cup BE \cup FROZEN$ has stopped being in this set. This can only happen by expanding s' before its successor along the path. But then, the successor is added to $OPEN$ and its g-value is made optimal by the expansion of s'. Therefore, the invariant is maintained. □

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

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

Fix an optimal path to s, and let s' be the first node on it which is in $OPEN \cup BE$. Now, there are two cases to consider.

If s' is one of the nodes which were originally transferred from $FROZEN$ to $OPEN$ in the current iteration of the main() loop, then $g(s') = g^*(s')$. Therefore,

$$g_p(s') \leq \epsilon g(s') = \epsilon g^*(s').$$

Otherwise, let s_p be the predecessor of s' on the optimal path. Then $s_p \in CLOSED$ so, by the induction hypothesis, $g(s_p) \leq \epsilon g^*(s_p)$. Therefore,

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

In either case,

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

Therefore, by Lemma 2,

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

□

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 PAPA* 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 thus follows from Theorem 1. □

Theorem 2. *If $w \leq 1$, the parallel depth of checkless PAPA* 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 PAPA* 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 checkless PAPA* 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.