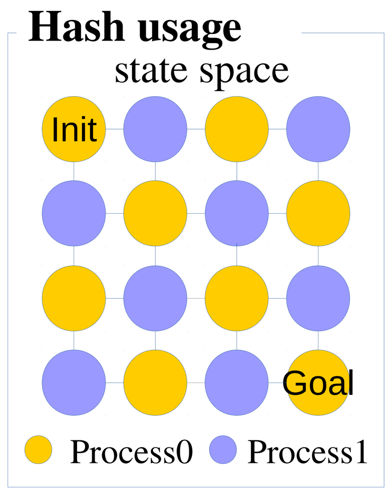
A Graph Partitioning-Based Work Distribution Method for Parallel Best-First Search

Yuu Jinnai and Alex Fukunaga (RIKEN, The University of Tokyo)

Static vs. Dynamic Load Balancing

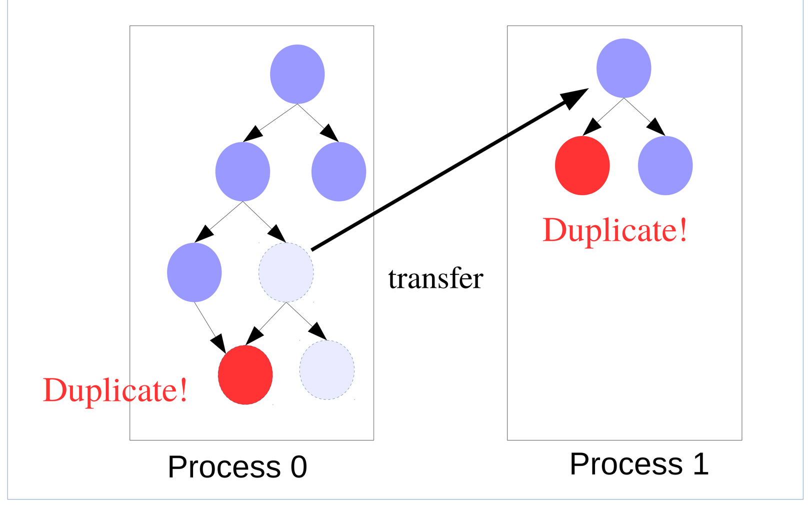
Static Load Balancing

Hash Distributed A* (Kishimoto et al. 2009)



Dynamic Load Balancing

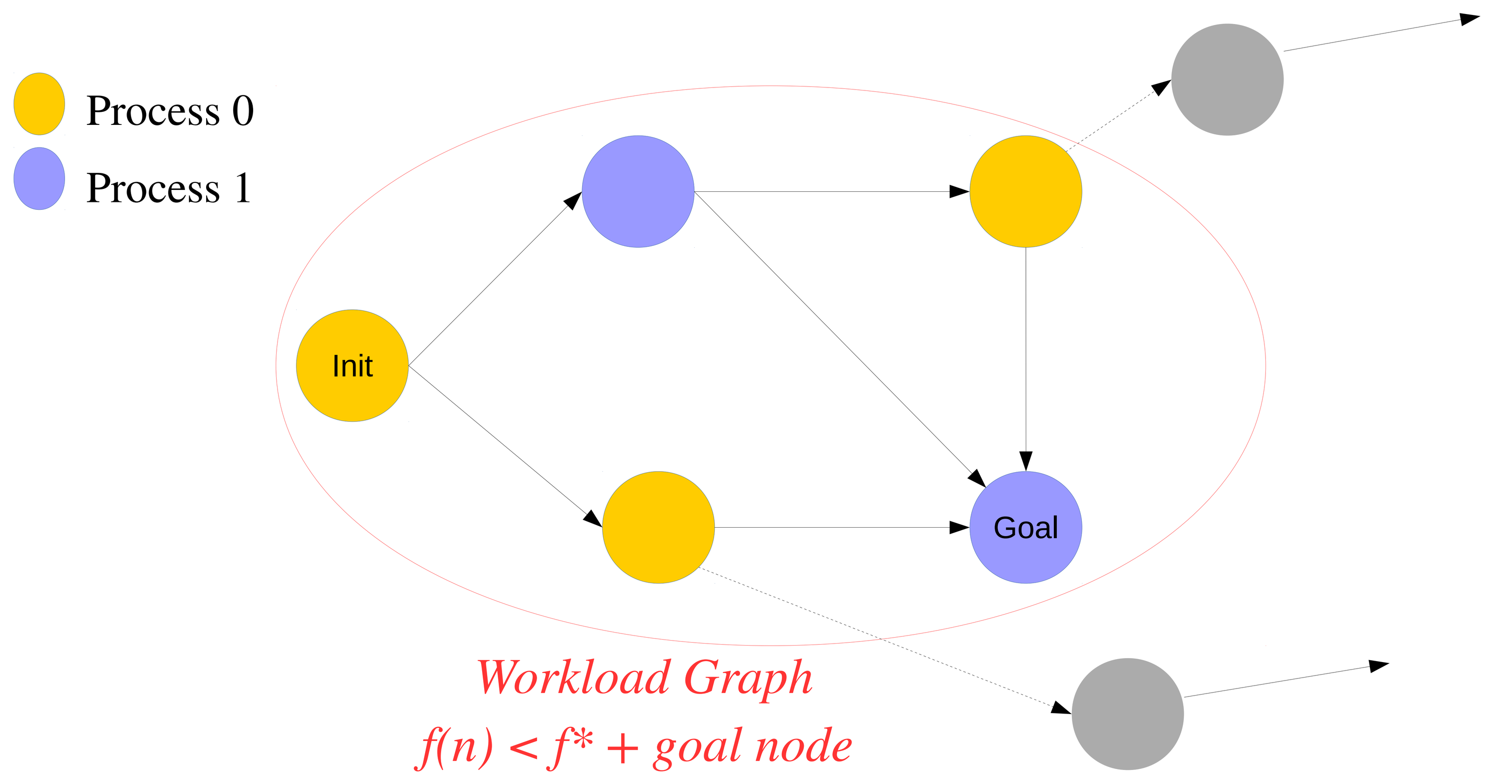
Work Stealing (Rao&Kumar 1987)



Problem of dynamic load balancing is that it generates potentially exponential **duplicated nodes** for graph search.

Hash Distributed A* (HDA*) is a parallel best-first graph search (A*) which distrbutes nodes according to a hash function which assigns each state to a unique process. As HDA* relies on the hash function for load balancing, **the choice of hash function is crucial to its performance!** However, **previous works relied on ad hoc tuning to achieve good performance**, and are not based on an explicit model which we can estimate the performance on.

Model of Parallel A* with Static Load Balancing



Workload Graph
$f(n) < f^* +$ goal node

1. Expand a node owned by the process ($t = t_{proc}$)
2. Send child nodes to their owner ($t = t_{com}$)
3. Terminates when all nodes are expanded and sent (to ensure optimality)

$$CO := \frac{\text{number of edges which require communication}}{\text{total number of edges}} = \frac{4}{6} = 0.66$$

$$LB := \frac{\text{maximum number of nodes owned by a process}}{\text{average number of nodes owned by a process}} = \frac{3}{2.5} = 1.2$$

- Communication Efficiency
  - Assume communication cost for every pair of processors are idential
  - The degradation of walltime effciency by communication

$$eff_c := \frac{1}{cCO} \quad \text{where} \quad c := \frac{t_{com}}{t_{proc}}$$

- Search Efficiency
  - The ratio of the increase of the number of nodes expanded compared to sequential search
  - The degradation of walltime efficiency by search overhead

$$eff_s := \frac{1}{1+SO} \quad \text{where} \quad SO = p(LB-1)$$

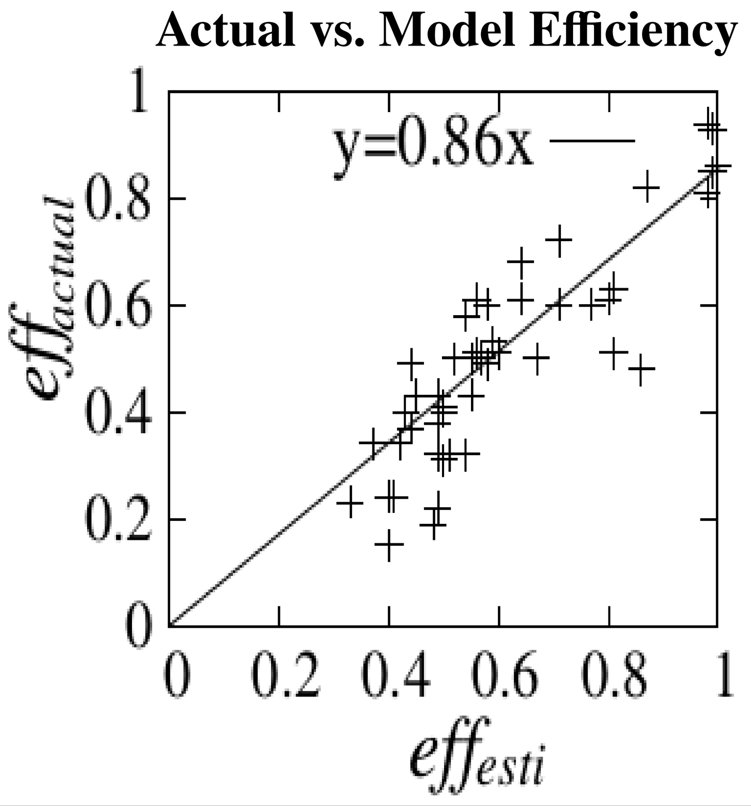
- Model Efficiency
  - Assume communication and search overheads are the dominant overhead

$$eff_{esti} := eff_c \cdot eff_s = \frac{1}{(1+cCO)(1+p(LB-1))}$$

Using CO and LB we can model the walltime efficiency of the Parallel A* on the graph

$$eff_{esti} := \frac{1}{(1+1\cdot 4/6)(1+2(3/2.5-1))} = 0.42$$

Experimental Comparison of Actual vs. Model Efficiency



Summary

Background: The state-of-the-art strategy of work distribution for parallel A* is **static load balancing**: assigning each state to a process by a global hash function. However, there was **no quantitative analysis** on what kind of hash function is optimal**.**
Main contribution: We deploy a model of parallel A* to examine the effectiveness of hash functions. Using the model, we propose graph partitioning-based approach for work distribution method. Our experimental results show that our method significantly outperforms previous methods.

Why Parallel Search?

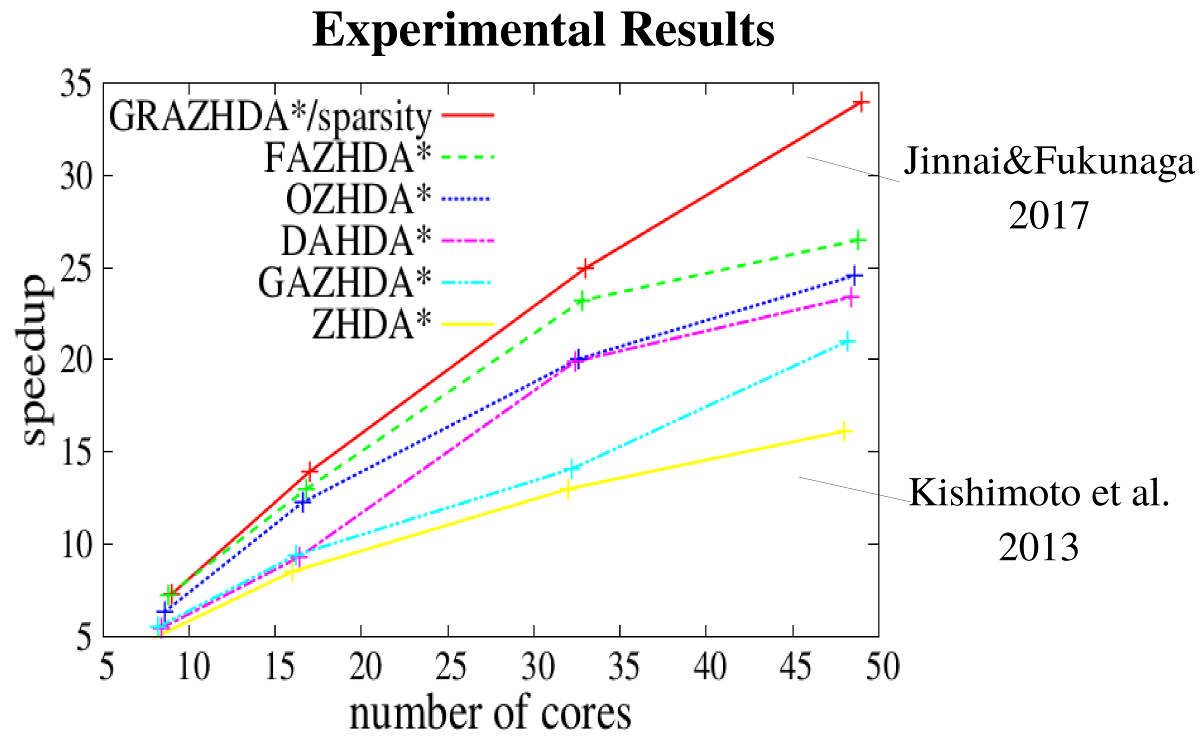
Both time and space are bottleneck of A* search. **Both can be addressed by parallel search on distributed environment.**

Supercomputer (TOP 500)
https://www.top500.org/statistics/perfdevel/

Experimental Results and Model Efficiency

- Evaluated on a 48 core cluster with 6 hashing functions
- merge&shrink heuristic

Experimental Results



Comparison of Model Efficiency