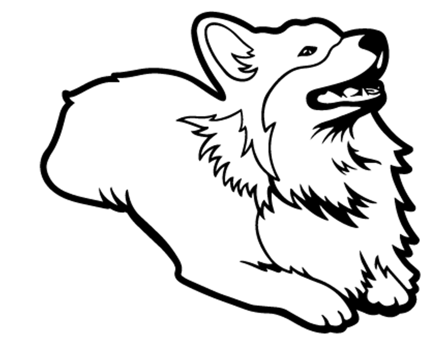
Harmony: Co-Optimizing Parallelism and Locality to Bound Performance



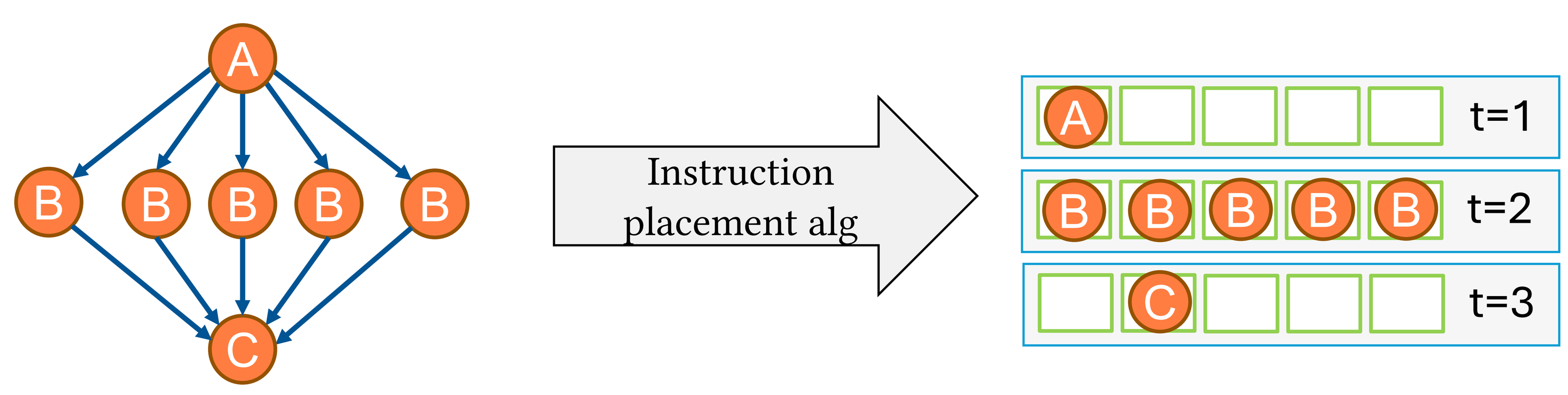
Jennifer Brana, Nathan Beckmann

Carnegie Mellon University

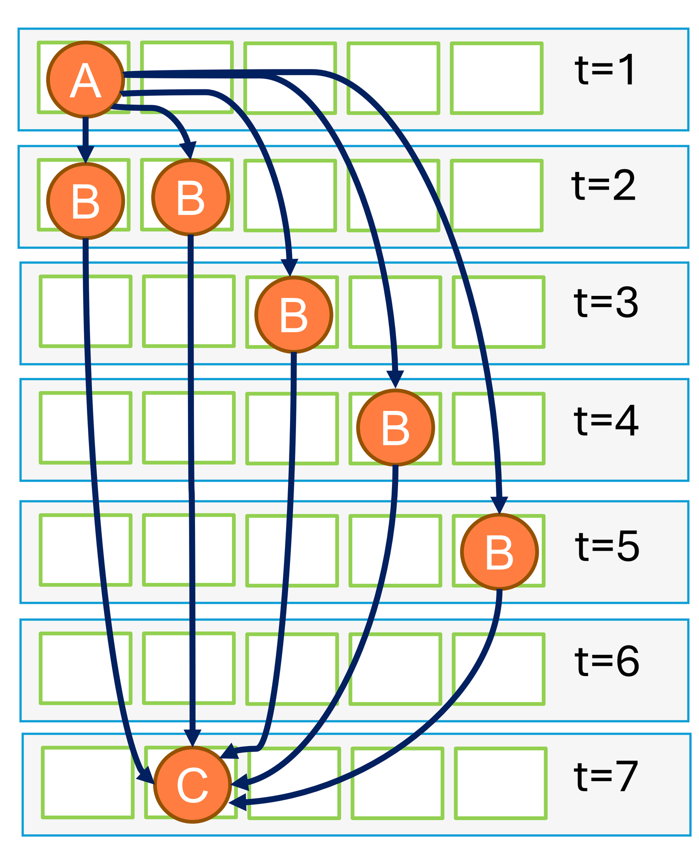
Data Movement-Aware Bounding

Our goal: understanding the fundamental limits of performance in real systems

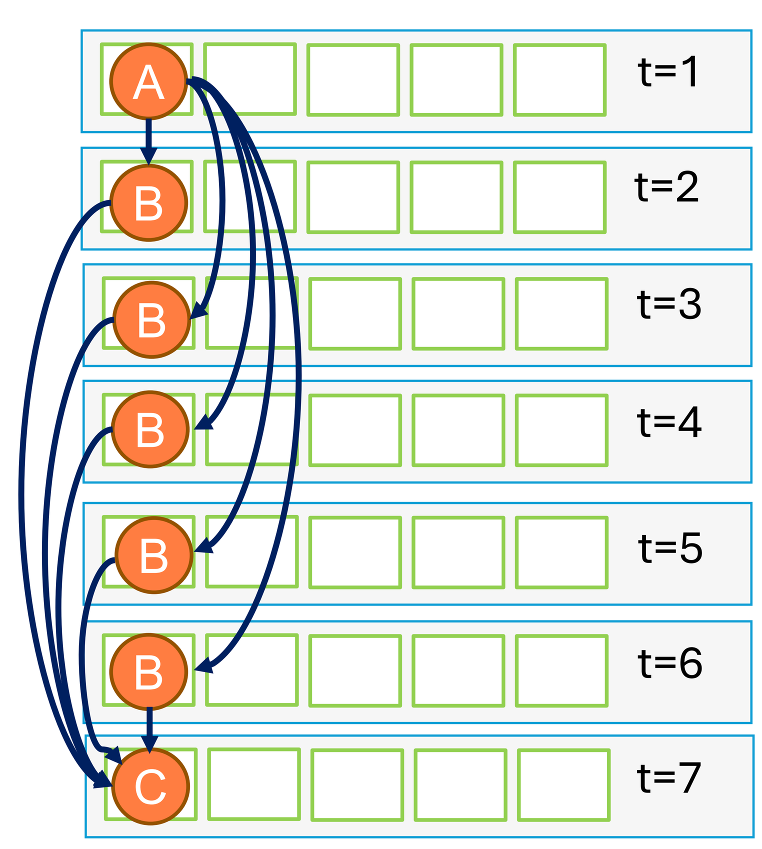
Our approach: instruction placement in parallel processors



Why instruction placement?

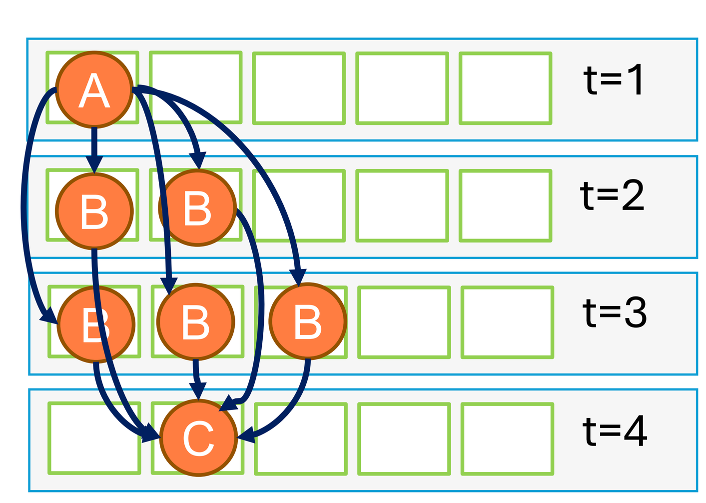


...captures the impact of communication latency & instruction fetch penalty



...captures the effect of exploiting locality to reduce data movement overheads

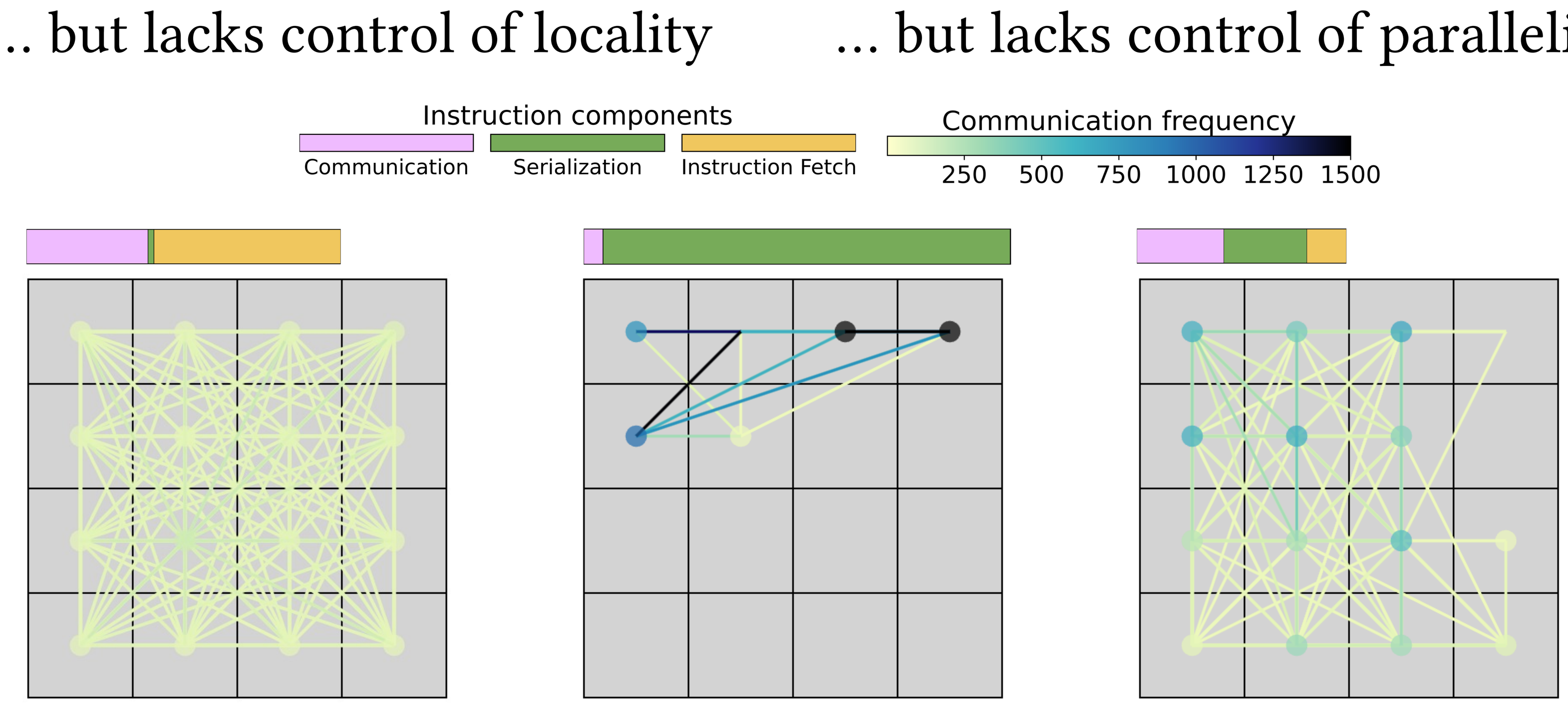
...enables us to exploit the tradeoff between parallelism and locality to maximize performance



Limitations of Prior Work

OoO superscalar models: Optimize parallelism via dynamic scheduling to expose parallelism within & across loops
... but lacks control of locality

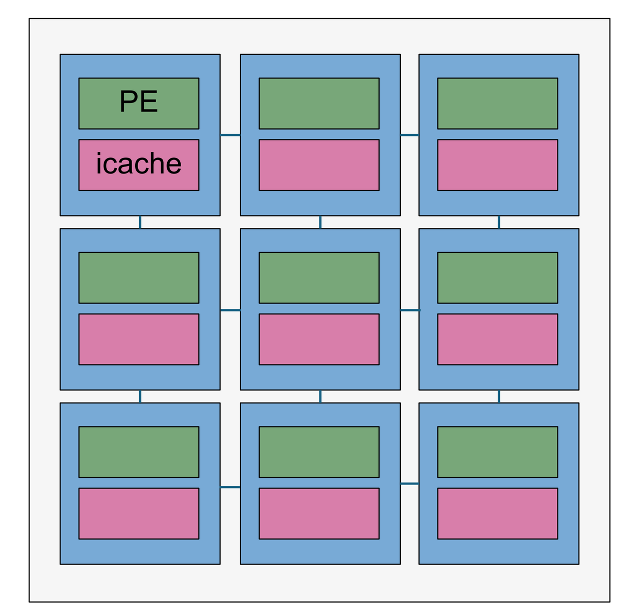
Spatial dataflow models: Optimize locality by unrolling program in hardware & binding static instructions to PEs
... but lacks control of parallelism



Random (OoO) Snake (spatial) **Harmony**

Chaos Processor Model

Chaos is a processor model that lets us isolate and study the tradeoff between parallelism and locality in a single framework



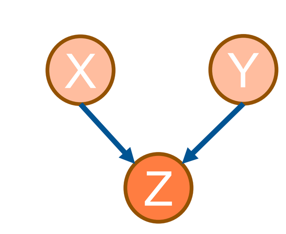
...incorporates **dynamism** of OoO scheduling to throttle parallelism

...a **spatial execution model** exposes control of dataflow and instruction locality to online instruction placement algorithms

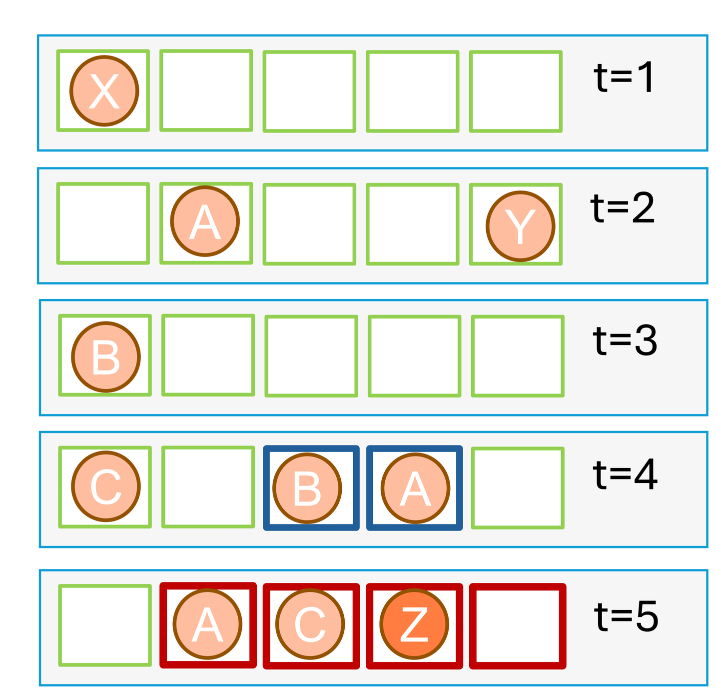
Harmony Placement Algorithm

Harmony places instructions by performing a local search over space and time, scheduling the instruction at the earliest available PE, according to locality and already-scheduled instructions

Harmony example placing operation 'Z':



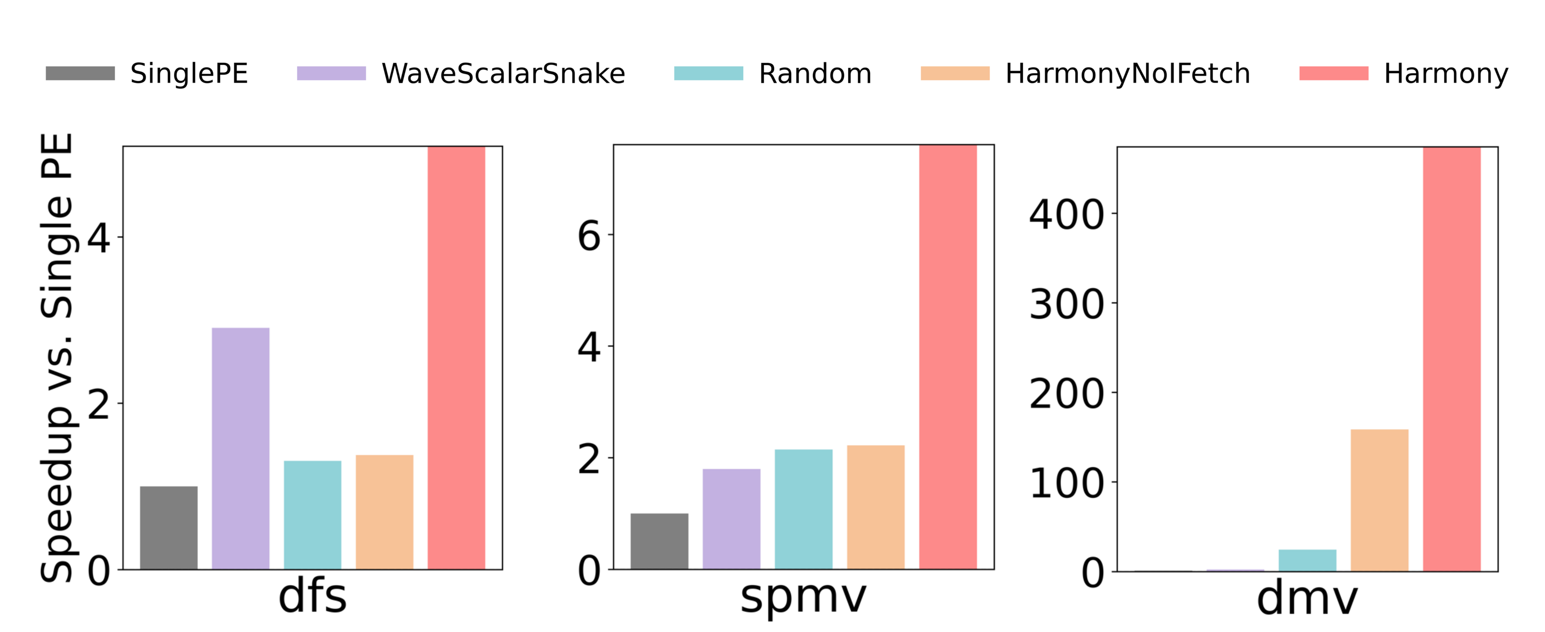
1) Find PEs reachable earliest by both input operands (x & y)
   - *Highlighted in blue*
2) Check when PEs can execute Z
   - *Other isns are scheduled on the ideal PEs*
3) If no PEs can execute Z at the earliest time, PEs one cycle from the list are added and the next cycle is checked
   - *Highlighted in red*
   - *Z is scheduled on PE 3*



Evaluation

We evaluate **Harmony** using a trace-driven simulation of **Chaos**

We evaluate 6 common workloads: DFS, BFS, SMV, SpMV, DMV, and DMM



Harmony averages a speedup of **1.98×** over the next best method

...co-optimizes parallelism, dataflow locality, and instruction locality

...finds the best balance between parallelism and locality

Carnegie Mellon University