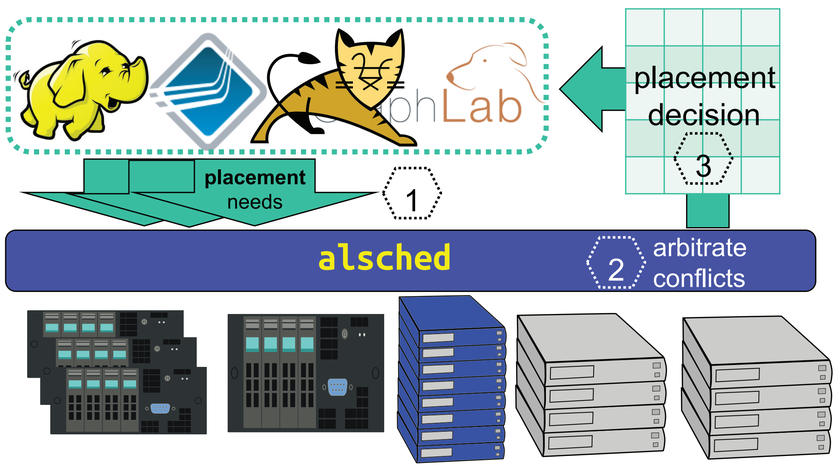
alsched: SCHEDULING OF MIXED WORKLOADS IN HETEROGENEOUS CLOUDS

Alexey Tumanov, James Cipar, Gregory R. Ganger (Carnegie Mellon University), Michael A. Kozuch (Intel Labs)

HETEROGENEITY AND SCHEDULING

- Large clusters shared by varied workloads
  - Batch frameworks, like Hadoop
  - Elastic services, like web frontends
- Cluster nodes increasingly heterogeneous
  - Static: amount of RAM, #cores, GPU?, …
  - Dynamic: cached executables, storage/net locality, …
- alsched: matching diverse needs to resources
  - User agents request desired resources
  - Utility functions used to express placement constraints
  - alsched arbitrates conflicts quantitatively

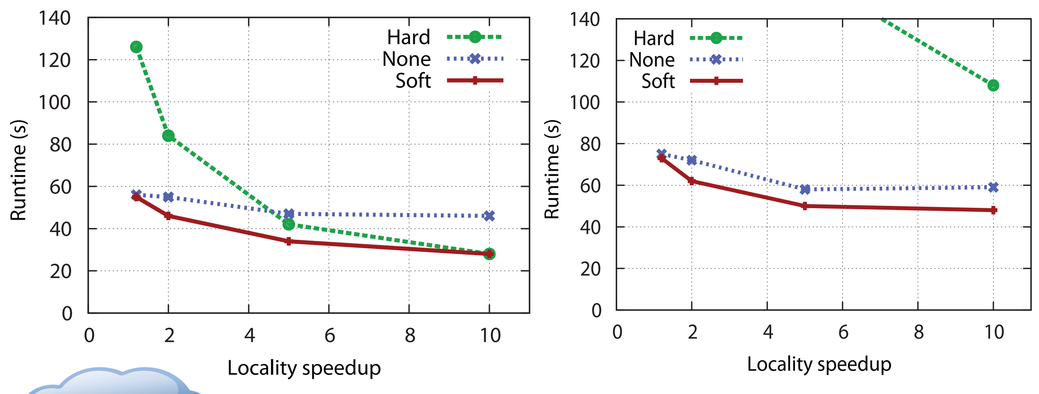


PLACEMENT CONSTRAINTS

- Mandatory or preferred task placement restrictions
  - Defined over machine attributes or subsets
- Hard constraints communicate requirements
  - Must avoid machines with attribute X
  - Must run on machines with attribute Y
  - Require kernel version > 2.6.35
- Soft constraints communicate preferences
  - Prefer machines with attribute X
  - Locality: prefer k tasks on same rack with infiniband (e.g., MPI)
  - Affinity: prefer to run close to data
  - Anti-affinity: prefer to spread tasks (e.g., for availability)

PRELIMINARY EVALUATION

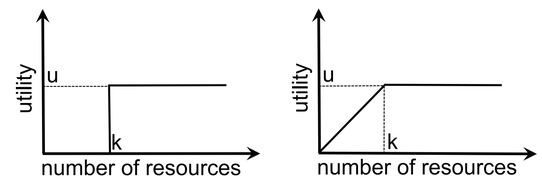
- Simulated workload of n-body type jobs
- Scheduling policies compared:
  - Soft – soft constraint-aware placement
  - Hard – soft constraints treated as hard
  - None – soft constraints are ignored



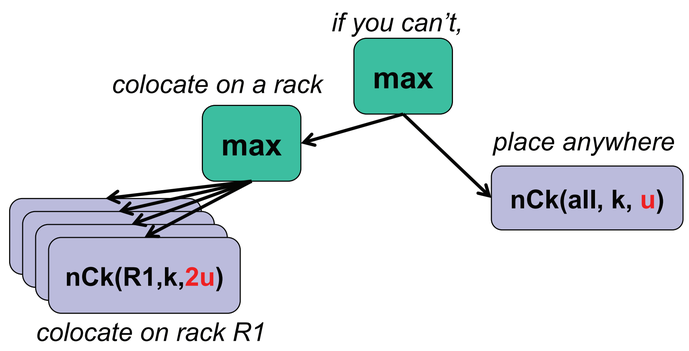
Intel Science & Technology Center for Cloud Computing

COMPOSABLE UTILITY FUNCTIONS

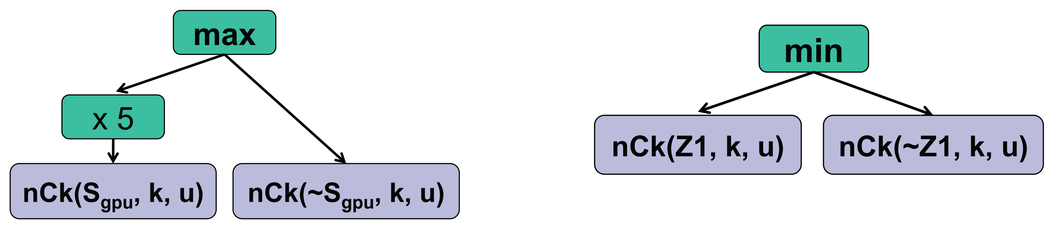
- Map resource subsets to utility values
  - Express both hard & soft constraints
  - Quantify benefits of preferences
- Primitives : “n choose k” , linear “n choose k”



- Operators:
  - Min/Max/Sum(u1, u2, u3, … un) → min/max/sum of its children
  - Scale(f,u) → f * u
  - Step(M, u) → M iff u ≥ M and 0 o.w.
- Examples:
  - Colocate k tasks on the same rack, else schedule anywhere

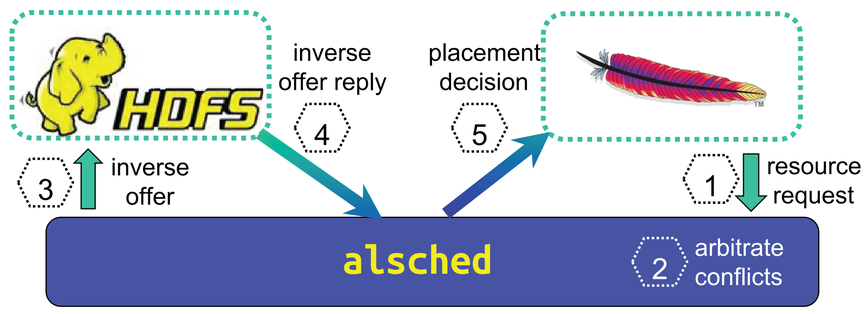


- Primary + backup service instance
- Specialized hardware



TWO-LEVEL CHANGE DECISIONS

- Involve cluster + framework schedulers
- Inverse offers: must give up X of Y where Z
- Use utility functions to guide inverse offers



CONTINUING RESEARCH

- Automatic generation of utility functions
- Handling imperfectly specified utility functions
- Handling placement change decisions
- Inverse offers

Carnegie Mellon University · Georgia Tech · intel · PRINCETON UNIVERSITY · UC Berkeley