Compiler Enhanced Scheduling for Heterogeneous Multiprocessors

Jyothi Krishna V S

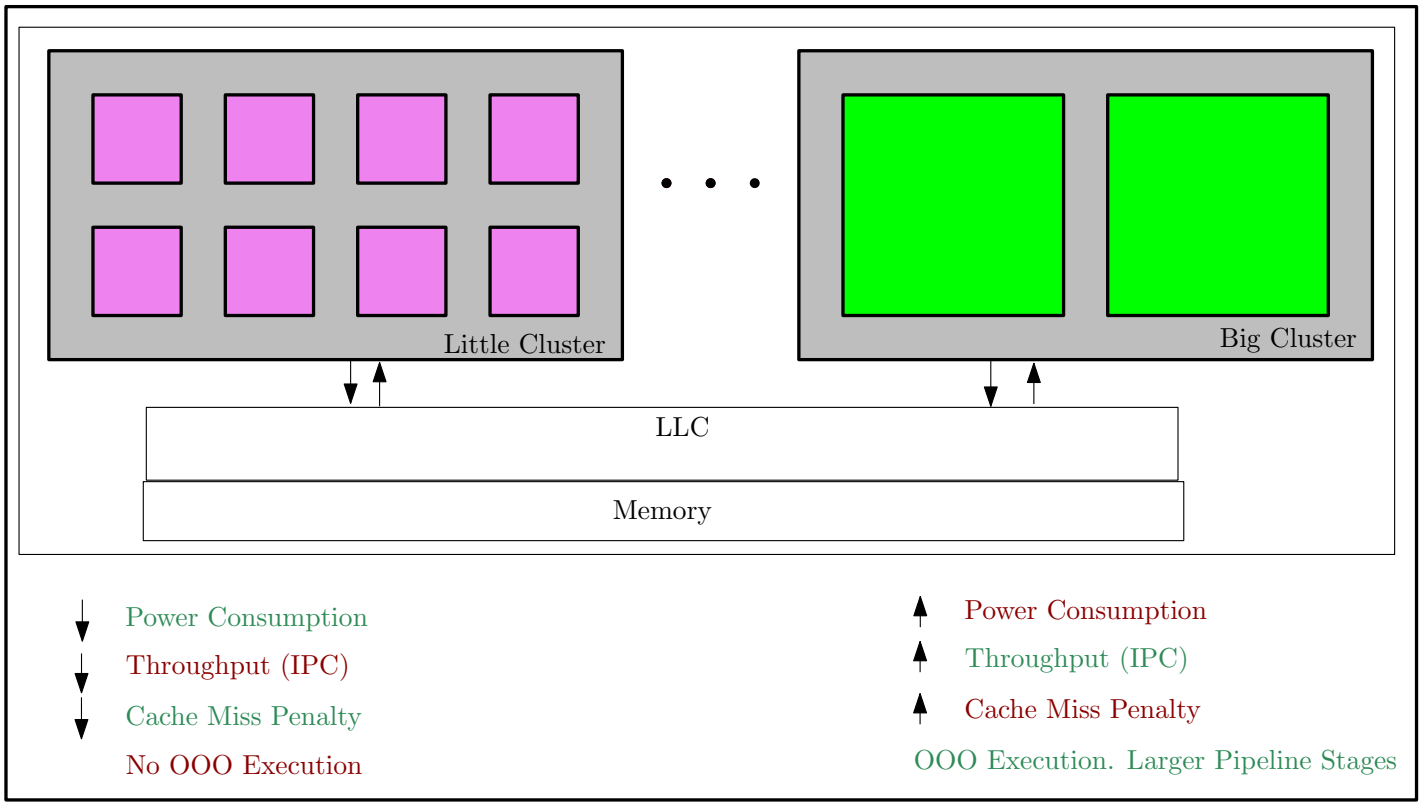
jkrishna@cse.iitm.ac.in

Advisors: Shankar Balachandran and Rupesh Nasre

IIT Madras

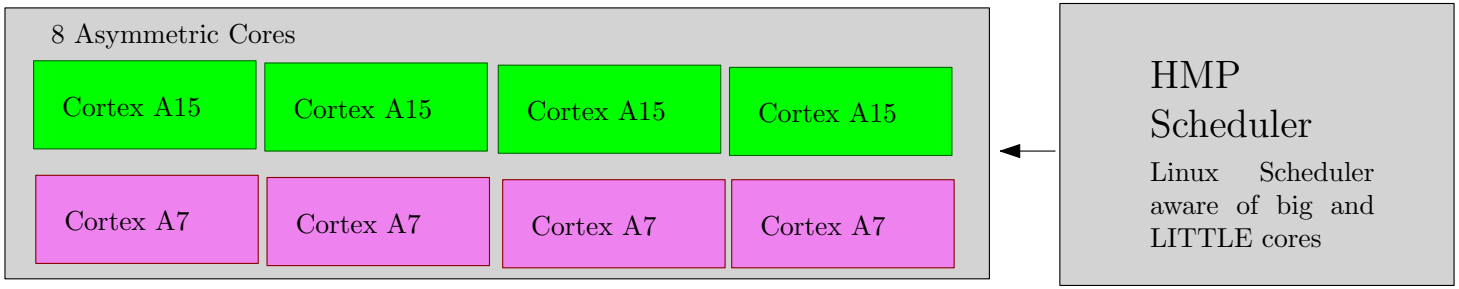
July 1, 2016

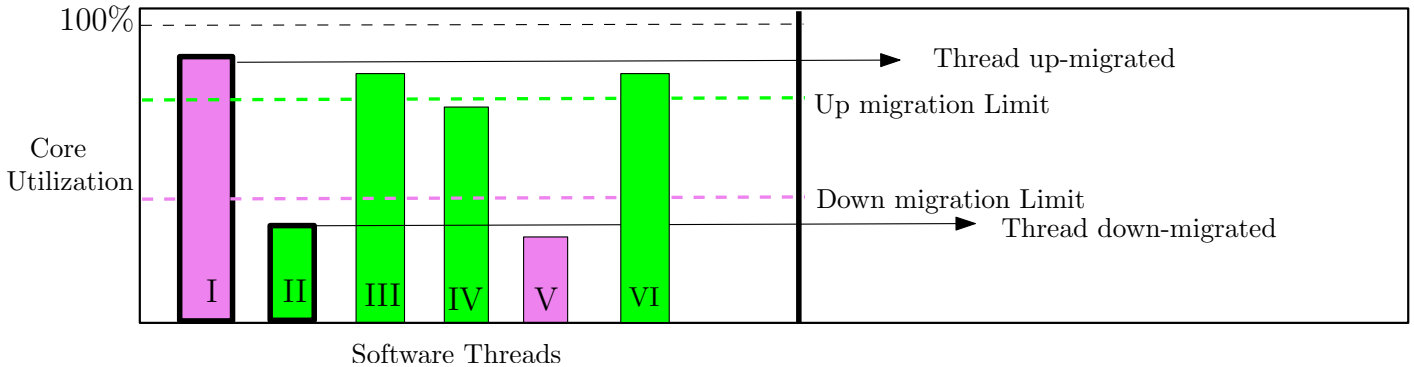
Asymmetric Multicore Processors



- Asymmetric Multicore Processors. e.g. `big.LITTLE`
- Different from NUMA Architecture: shared LLC.
- Compiler unaware of hardware asymmetry.

Compiler Scheduling and HMP





- HMP: Heterogeneous Multiprocessor Scheduling.
- OpenMP: Optimized for SMP
- `T (parallel unit)`: One task or piece of code that can run in parallel with others.
- e.g.: one iteration in `omp parallel for`, `omp parallel section`

Knapsack

- Cost function: the run-time objective
  - e.g. Improve run-time T_i^c = Expected run-time of T_i.
  - e.g. Improve Energy : EDP etc.
- Modified Multi Knapsack problem.
- Modification: Each object has different cost when put into different sack.

Capacity: $\sum_{i=0}^{N} WC(T_i)/n(Cores)$

Where, $WC(T_i) = max(T_i(b), .., T_i(l))$

Cost of placing T_i in a sack, b:

$T_i^c(b)$ = Estimated cost of running T_i in core b

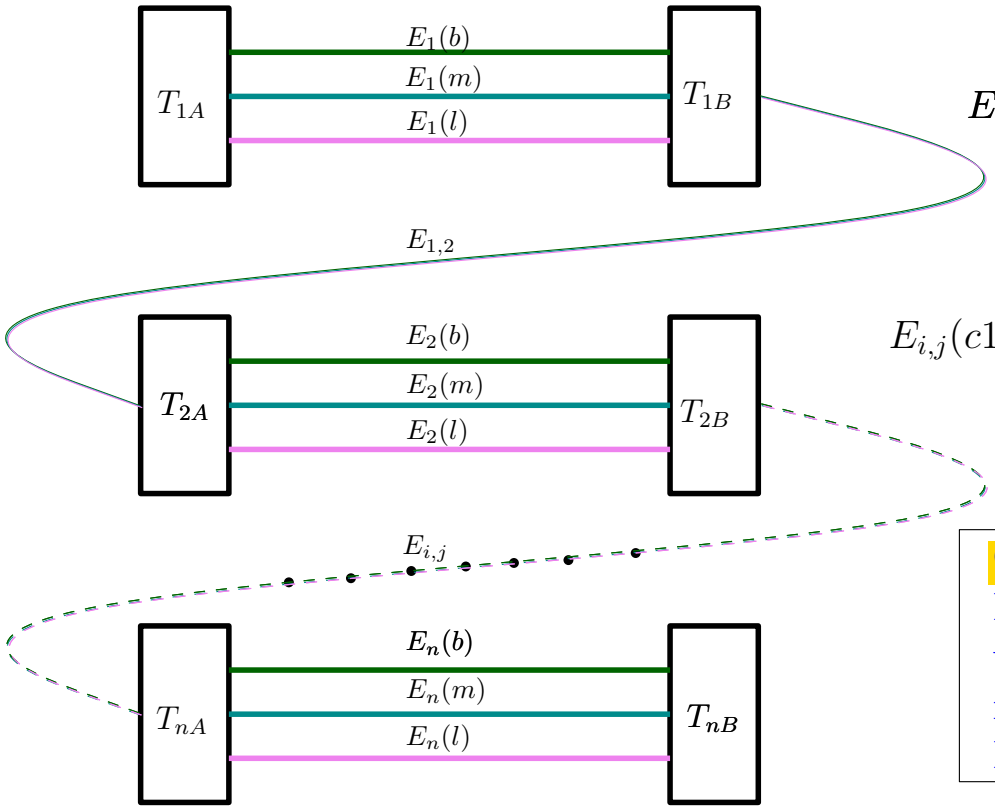
Profit on placing T_i in a sack, b:

$T_i^p = E(T_i^c)$

Maximize profit, with minimal cost

Minimizing Colored Edge Weight

- The maximum weight of each color: Capacity in Knapsack.
- Optimizes for cache locality.
- Task Edges and Correlation Edges.
- Choose only Task Edges.



T_{iA} - T_{iB}: T_i

$E_i(x)$: Cost of executing i in Core x

Task Edge.

Correlation Task

$E_{i,j}(c1, c1) = 0$

$E_{i,j}(c1, c2)$: Correlation cost of T_i and T_j

$E_{i,j}(c1, c2) > 0$ when $c1 \neq c2$

Objective

Connected subgraph with lowest Edge weight such that the total weight of a single color is minimized. Retain all correlation Edges.

Conclusion

- Source to source transformation.
- All parallel constructs are converted into worklist based model.
- OpenMP Tasks: recursion ??
- Providing handles for users ??

Run in big only

```
#pragma omp for schedule(bigonly)
for(i=0;i<n;i++){
   [...]
}
```