Università della Svizzera italiana | Faculty of Informatics | Institute of Computational Science ICS

Graph Partitioning with OpenMP Task Parallelization

Rodrigo P. Coelho, Università della Svizzera italiana, Lugano, Switzerland

The Graph Partitioning Problem

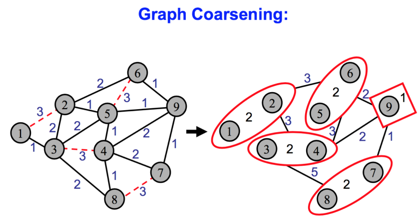
Given:
A graph $G = (V, E)$ and number of partitions, p

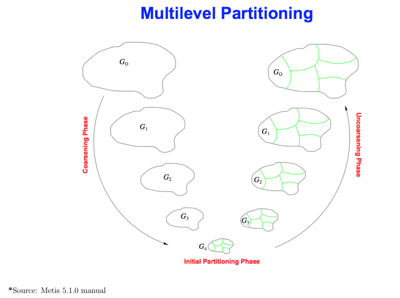
Output:
A (vertex) partition $V = V_0 \cup V_1 \cup \ldots \cup V_{p-1}$

Such that:

1. $\{V_i\}$ are disjoint $\Rightarrow V_i \cap V_j = \emptyset$
2. $\{V_i\}$ are roughly balanced $\Rightarrow |V_i| \approx |V_j|$
3. Let $E_{\text{cut}} = \{(u, v) | u \in V_i, v \in V_j, i \neq j\}$
   $Minimize : |E_{\text{cut}}|$

Multilevel Partitioning Algorithm:





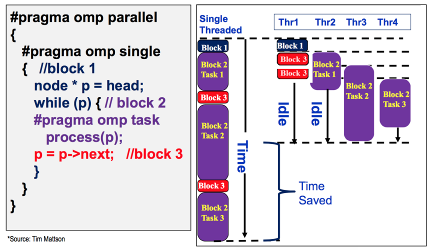
Tasks in OpenMP

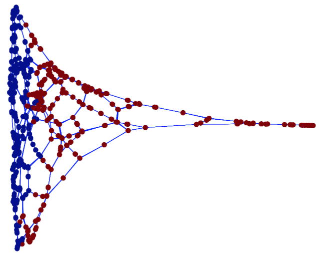
Independent units of work:

Composed of:

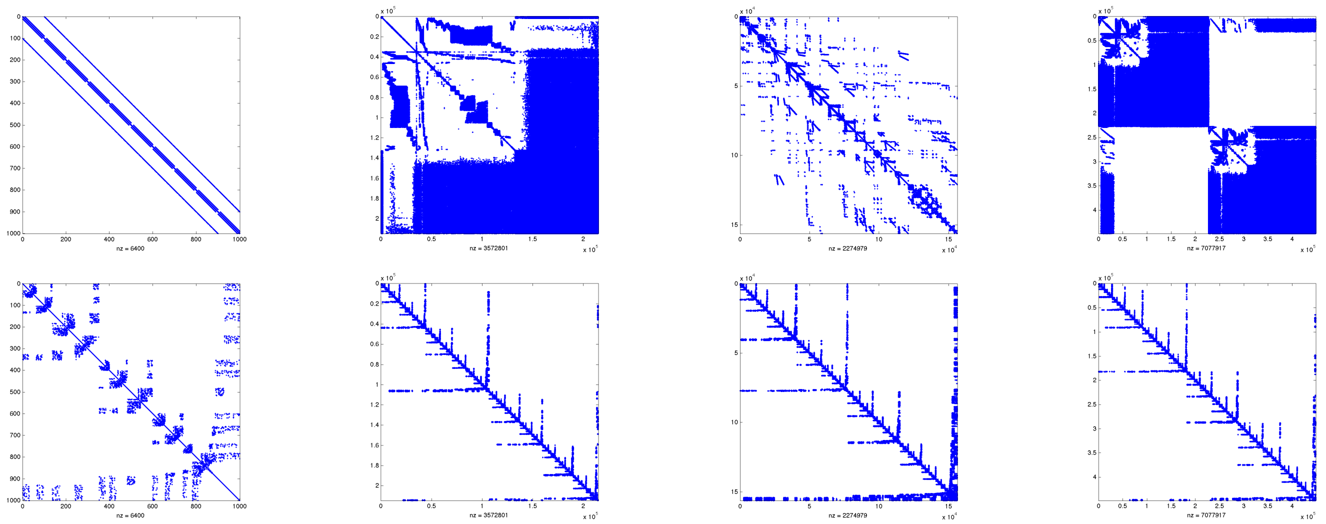
- Code to execute
- Data environment
- Internal control variables (ICV)

- Threads perform the work of each task.
- The runtime system decides when tasks are executed.
- The task directive defines the code associated with the task and its data environment.





Reordering



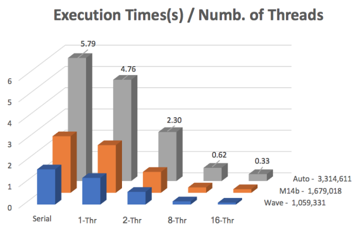
Parallel x Serial Performance

The performance evaluations were run on 2 x Intel E5-2650 v3, 16 (2 x 8) cores with 64MB of DDR4 @ 2133MHz RAM. The graphs for the experiments were obtained at the University of Florida Sparce matrix collection (www.cise.ufl.edu). The graph names, sizes and execution times are described in the next panel.

As can be observed, speed ups of about 17 times on average were reached on our hardware set up. The speedup achieved is largely dependent upon the size of the graph and how well the graph is distributed among threads as each of them complete the tasks on their assignment list and start helping other threads.

Name	Vtxs	Edges	Serial	2-Thr	8-Thr	16-Thr
auto	448,695	3,314,611	5.79	2.30	0.62	0.33
m14b	214,765	1,679,018	2.66	0.98	0.25	0.17
wave	156,317	1,059,331	1.65	0.59	0.15	0.09

Table 1 - Execution Times in Seconds



References

[1] D. LaSalle and G. Karypis. A parallel hill-climbing refinement algorithm for graph partitioning. In *2016 45th International Conference on Parallel Processing (ICPP)*, pages 236–241, Aug 2016.

[2] D. LaSalle and G. Karypis. *Efficient Nested Dissection for Multicore Architectures*, pages 467–478. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[3] A. J. Soper, C. Walshaw, and M. Cross. A combined evolutionary search and multilevel optimisation approach to graph-partitioning. *J. of Global Optimization*, 29(2):225–241, June 2004.