Computer Science Terms Explanation

Athanasio, Chourlias
it219113@hua.gr

Nikolaos, Liapis
it21950@hua.gr

April 25, 2022

1 Task Dependency Graph

Before we define and describe what is the task dependency graph we should take a step back and start by defining what we mean with the word task. A task is defined as the unit of computation which is programmer defined, such computational units(tasks) come from task decomposition techniques, which we are not going to explain or examine in this simple one subject explanation paragraphs.

Firstly we need to the definition of the term 'task dependency graph'.According to bibliography, dependency graphs and as a result task dependency graphs are defined as a directed graph G=(V,E) where V(tasks) is a set of nodes and $E \subseteq$V$\times V$is a set of edges that represent control dependencies, while allowing self-edges[1].Now, since we have defined task dependency graphs let us explain the where and how they are useful.In general, dependency graphs have many applications in many computer science fields and research areas, two of the most widely known are computer graphics and neural networks, on the other hand now task dependency graphs are commonly used in compilers, schedulers e.t.c, such an example would be a scheduler that uses a task dependency graph in order to ensure that work is equally distributed across all processors at any given point, with the benefit of minimum idling and optimal load balancing[3].

2 Granularity In Parallel Computing

First and foremost, it is necessary before we write down any definitions to explain and understand what granularity means when we refer to it in parallel computing. The term granularity, meaning size, of units of work in parallel computing is the number of sub-units of work which a problem is decomposed to[3]. Although, we will focus on a more specialized definition of the term granularity in parallel computing which describes the overhead in between hardware components(Processors) and tasks(Processing elements). Now that we have a better understanding of what granularity is, let's examine the definition mentioned before. The granularity of a parallel computer is defined as the ration of the time that is required for a basic communication operation to the time required for a basic computation operation[5].

3 Embarrassing Parallel Workload

With the term embarrassing parallel workload, we mean that a problem(workload) can be decomposed down to fine-grained tasks(units of work)very easily[2].One major characteristic of embarrassingly parallel problems is that when the workload is finally divided in a number of parallel tasks, these tasks do not have any dependencies between them or if any this happens rarely and the dependency is negligible.Furthermore, these types of tasks don't have any need to communicate between them. Also if we look close enough we will notice that by having independent tasks without the need for communication running in parallel, then the produced results will also be independent. Last but not least, we need to mention that the opposite of embarrassing parallel problems are inherently serial problems, where there is complete dependency between of the tasks[4]. Every task needs the result of the previous in order to complete.

References

[1] Predicting defects using network analysis on dependency graphs.

[2] Best practices for executing embarrassingly parallel workloads with R Server on Spark, March 2019. Section: SQL Server Blog.

[3] Ananth Grama, Vipin Kumar, Anshul Gupta, and George Karypis. *Introduction to Parallel Computing*. Pearson Education, 2003. Google-Books-ID: B3jR2EhdZaMC.

[4] Sujanavan Tiruvayipati and Ramadevi Yellasiri. Practicability of embarrassingly parallel computations for enormous miniature workloads over massive underutilized IoT. In *2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)*, pages 1–4, November 2019.

[5] Roman Wyrzykowski, editor. *Parallel processing and applied mathematics: 4th International Conference, PPAM 2001, Nałeczów, Poland, September 9-12, 2001: revised papers*. Number 2328 in Lecture notes in computer science, Lecture notes in artificial intelligence. Springer, Berlin ; New York, 2002. Meeting Name: PPAM 2001.