Algorithms and parallel programming: Problem E

Alejandro Molina Zarca – alejandro.molina2@um.es
Computer Science Faculty – University of Murcia, Spain

1. Abstract

This document describes the method used to solve problem E featured in practices of Algorithms and Parallel Programming course.

2. The problem statement

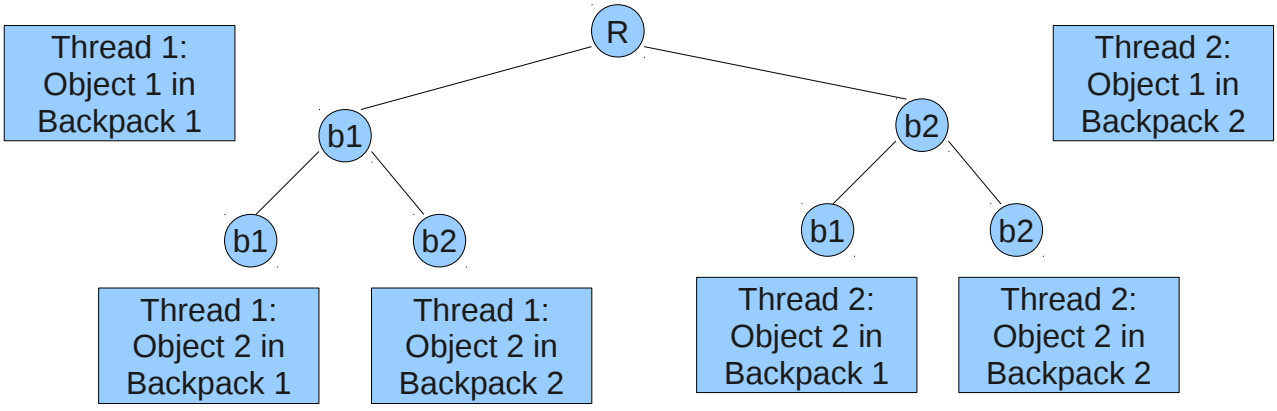
We have several knapsacks, each with a certain capacity, and some objects with affinities (the similarity between two objects is the benefit gained by putting them together in the same bag) and a certain weight. The aim is to assign objects to knapsacks, taking into account the weight of the objects and the capacity of the knapsacks, which maximizes the sum of the benefits of all the knapsacks, where the benefit of each knapsack is obtained by adding together the affinities between objects.

3. Solving the problem

To solve the problem, differentiate the following cases:

1. Number of cores < number of knapsacks
2. Number of cores = number of knapsacks
3. Number of cores > number of knapsacks

In the first and second cases, generate sequentially resulting nodes to put the first item in each knapsack.



In the last case, it is necessary to generate sequentially enough nodes to provide work for all cores. A generator of nodes is used up to level “n”, where “n” is a level that claims to have at least as many nodes as cores. When an invalid node is generated (any weight exceeds the knapsack) it is discarded.

When the nodes have been generated, the master process sends as many nodes as possible threads have been established. If there are not as many nodes as threads, send as many as possible and add an end mark. When a slave receives the nodes, it seeks the best solution from each node and returns the best. The master process stores the best solution from all those received. When there is no more work, the master process sends an end mark to slaves.