Results and programming techniques from the CR3x+1 project

G. Scollo

Consorzio COMETA - Dipartimento di Matematica e Informatica, Università degli Studi di Catania, Italy

http://www.dmi.unict.it/~scollo

Introduction

The CR3x+1 project runs a parallel algorithm on the Cometa Grid Infrastructure for the search of Class Records in the 3x+1 Problem. The search up to 2^{59} was recently completed, following a 15-month computational effort that delivered 150 new Class Records. Besides presenting these results, a few techniques adopted in this project are introduced that are aimed at keeping the search proceed on schedule despite of occasional failures and delays of running jobs. The ideas behind these techniques seem easily adaptable to a wider class of search programs over a countable search space. Examples of problems solved by this kind of programs are provided. A forthcoming paper will make an attempt to characterize such a wider applicability class, will provide a summary of the measured performance of the CR3x+1 search program, along its evolution throughout several running versions, and will outline a few current research questions.

3x+1 Class Records search

The "$3x+1$" function on the positive natural numbers is defined as quoted for odd x, while it halves the even numbers. The iterates of this function seem to eventually converge to (a loop through) the number 1, but no proof of this conjecture has been found to date. Numbers are classified by the delay (number of steps) of their $3x+1$ trajectory to reach 1. A *Class Record* (CR) is a smallest number in its delay class. A computational challenge is that CR's may only be found by *exhaustive exploration* of an infinite search space. An inventory of known *optimization techniques* provides savings on the computational cost of the CR search.

CR3x+1 progress and results

15-month CR search in the COMETA GRID Infrastructure (9/2007 – 12/2008) delivered 150 new Class Records and a tenfold widening of previously explored search space, thus covering all numbers up to 2^{59} (to date this has been raised up to $6{\cdot}10^{17}$ and 5 new CR's have been found).

Uncertainty and partial failures

Practical problems with Grid computing:

- occasional failures and delays of running jobs
- uncertainty about availability of resources (CPU's)
- partial reliability of job status information, etc.

Recursive parallelization

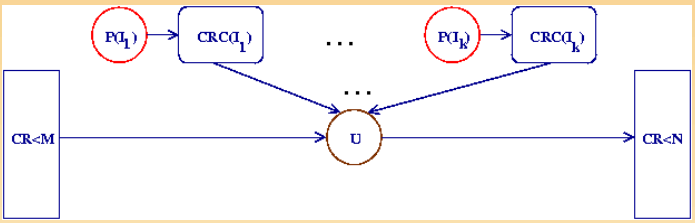
Job control and parallelization prove effective to cope with the first two aforementioned problem kinds, for:

small is easier (to get Done (with Success) ;)

that is: if a job fails, rather than restarting it, reassign its computational task to a collection of parallel jobs:

- if these are small enough, in the COMETA Grid many more CPUs are available
- small jobs have much higher chances to succeed

The basic CR search algorithm deployed on the COMETA Grid is easily amenable to DAG parallelization, based on search space partitioning, where independent, noncommunicating processes explore separate intervals. They produce *CR candidates*, while a *merge* process combines their outcomes together with previously found CR's, as depicted in the figure.



Recursive parallelization simply is the iterated application of this mechanism to missing search intervals because of job failure or excessive delay. A search taking a few days by a single job will take a few hours by a collection of, say, 64 parallel jobs, and if some of these fails, by recursive parallelization the missing tiny bits of the planned recovery will only take a few minutes to be filled up.

Wider applicability examples

The *reusability problem*: abstract from the specific $(3x+1)$ Problem at hand, to make our Grid programming techniques applicable to a wider problem class. Here are a few examples of problems in this class:

- search of unknown prime numbers;
- search of unknown twin prime pairs;
- CR search for classes of (consecutive) prime gaps (this problem is connected to the Goldbach conjecture).