Parallel Markov Clustering for Large-scale Protein-Protein Interaction Networks using MPI

Alhadi Bustamam$^{1,3}$  Muhammad Shoai Sehgal$^1$  Simon Wong$^1$
a.bustamam@imb.uq.edu.au  s.sehgal@imb.uq.edu.au  simon@simonwong.org

Mark A Ragan$^1$  Kevin Burrage$^{1,2}$m.ragan@imb.uq.edu.au  k.burrage@imb.uq.edu.au

1 ARC Centre of Excellence in Bioinformatics, Institute for Molecular Bioscience, The University of Queensland, St. Lucia, Brisbane, QLD 4072, Australia
2 COMLAB, University of Oxford, UK
3 Department of Mathematics, University of Indonesia, Depok, 16424, Indonesia

Abstract

Empirical study of networks has enlightened our understanding of many application domains including the topology of biological systems. The Markov clustering algorithm (MCL), originally developed for clustering graphs, has been adopted to clustering protein-protein interaction (PPI) networks. MCL is also becoming an effective algorithm for the extraction of complexes from interaction networks. However, increasingly large amounts of data and massive networks will lead to a sparse and very complex network structure. It is impossible to perform computations on networks with millions of nodes without specialized computer facilities. Here we introduce parallel implementation approaches to improve the performance of MCL, and to allow the analysis of very large PPI-network datasets.

Keywords: MCL, MPI, PPI networks, biological networks, clustering, advanced computing

1 Introduction

Recently the Markov clustering algorithm (MCL) [3], developed for clustering graphs, has been broadly adopted in bioinformatics applications, such as in large scale detection of protein families (Tribe-MCL) [4]. MCL is also effective in the extraction of complexes from interaction networks [1]. However, as the datasets become ever larger and more complex, advanced computing strategies are increasingly necessary [2]. In this poster, we present results showing improvements in performance of the MCL algorithm in parallel and distributed computing environments using Message Passing Interface (MPI) [6] and Fortran 90. We test the efficiency of our approaches on a variety of data sets including small, medium and large PPI data sizes (see Table 1) from BioGRID (http://www.thebiogrid.org/) and from multi-proteome comparisons generated within our groups [5].

2 Method and Results

We have investigated two parallelization approaches, such as Master-Slaves (MS) and Collective Communication (CC) methods, to implement a parallel MCL especially in the largest computation phases of the original MCL algorithm (the expansion and inflation phases). These implementations included single-vector passing MS (sMS), block-vector passing MS (bMS), full-block CC (fCC), and partial-block CC (pCC). From our investigation we found that CC, especially fCC, is very much superior to MS (for example, see figure Fig. 1). But we still need more investigation, especially with pCC, in order to deal with the storage and communication time issues in very large datasets.
Table 1: Datasets for test cases

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data Source</th>
<th>#Proteins</th>
<th>#Interactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small (SM)</td>
<td><em>D. Melangoster</em></td>
<td>BioGRID</td>
<td>5,156</td>
<td>51,050</td>
</tr>
<tr>
<td>Medium (SF)</td>
<td>All organisms</td>
<td>BioGRID</td>
<td>23,175</td>
<td>137,104</td>
</tr>
<tr>
<td>Large (SH)</td>
<td>Homology PPIs</td>
<td>UQ-IMB</td>
<td>125,008</td>
<td>25,990,126</td>
</tr>
</tbody>
</table>

3 Discussions

Full-block CC exhibits very good efficiencies - about 80% on all these datasets over a wide range of processors. This is excellent behavior, see Fig. 2. Moreover, fCC shows a speedup several orders of magnitude better than those of other methods we examined (for example, see Fig. 1). We were able to extract clusters from a dataset with 125008 vertices (25,990,126 edges) in only 25 minutes using fCC on 16 processors of HPC computer Cyclone (SGI Altix Bx2) at the University of Queensland. With a serial computer this would have taken at least 5 hours.

References


