Improved Randomized Upper Bound for Conflict Detectors

Computer Science 490 Project Description

Michael Kurland
Advisor: James Aspnes
February 1, 2012

1 Background

Conflict detectors are a class of objects that support “a single operation, check(v), with input v from a set of m values” [2]. This function returns true to indicate a conflict or false to indicate that there are no conflicts. If all check operations have the same input value, then all of these operations must false. If there are two operations with different input, then they return true [2]. Aspnes and Ellen show how to build these objects from adopt-commit objects and give upper and lower bounds (described below). The aim of this project will be to develop an algorithm to improve on the upper bound of these objects. Since with anonymous processes the upper and lower bounds match, the improvement will have to be made either taking advantage of identities or only be in a special case such as when there is a conflict to detect.

1.1 Adopt-Commit

An adopt-commit object, also known as a ratifier, is an object that can be used to achieve agreement and consensus [2]. An m-valued adopt-commit object is defined as an object supporting the operation adoptCommit(u) where u is an input value from a set V of m values. There are two potential results of this operation: (commit, v) or (adopt, v). Here v ∈ V, and commit and adopt represent the decision bit that
indicates whether a process should immediately commit to $v$ or simply just adopt it as its preferred value in later rounds of the protocol.

Aspnes and Ellen give a “matching upper and lower bound of $\Theta\left(\min\left(\frac{\log m}{\log \log m}, n\right)\right)$ for the space and individual step complexity of a wait-free $m$-valued adopt commit object that uses multi-writer registers for $n$ anonymous processes” [2]. This lower bound holds for both randomized and deterministic algorithms. For non-anonymous systems, however, their upper and lower bounds do not match, with a lower bound of $\Omega\left(\min\left(\frac{\log m}{\log \log m}, \sqrt{\frac{\log n}{\log \log n}}\right)\right)$ and an upper bound of $O\left(\min\left(\frac{\log m}{\log \log m}, n\right)\right)$. My project will aim to improve one of these bounds or find a better bound for certain cases, such as when there is a conflict to be detected.

1.2 Winnow Object

A winnow object uses a randomized algorithm to split a set of processes into two groups, one of which will be small with high probability [1]. With this object, there are $r$ rounds and in each round process $p_i$ has probability $\pi_r$ of writing and $1 - \pi_r$ of reading from an atomic register. If the process writes then it passes through to the next round, while it only passes through to the next round after a read if no other process has performed a write yet. This guarantees that at least one process will advance to the next round, and by carefully selecting $\pi_r$ based on the the number of processes participating in round $r$, the expected number of processes remaining in round $r + 1$ can be optimized.

2 The New Strategy

The new strategy that I will use in my project to improve the upper bound previously developed by Aspnes and Ellen [2] is to first use multiple winnow objects to knock down the number of potential values and competing processes and then to feed these remaining processes into an adopt-commit object. Since a conflict-detector can be created from an adopt-commit object [2] this can be used to create an improved conflict-detector. Aspnes and Ellen [2] show a lower bound (which is described in section 1.1) applies to both anonymous randomized algorithms in addition to deterministic ones for solo executions where there is exactly one process running or all processes have only one input. Therefore, the improved upper bound would be restricted to either exploiting identities or giving improved performance only when there is a conflict to detect.
3 Deliverables

The deliverable for this project would be a written report containing my work on improving the randomized upper bound for conflict detectors. This report would start with background on the problem and descriptions of the various objects considered to build this conflict detector. These objects will be stuck together in an attempt to improve the upper bound. The report will include a thorough description of my efforts to develop an improved upper bound and complexity analysis of proposed algorithms. Of course, if one of these efforts is successful then the report will focus on this improved algorithm. Finally, the report will include suggested future work that could be done to further improve or expand the upper bound that I develop, or if I fail to find a better upper bound, suggested paths that may lead to such a bound.

References
