**CSC 2231 Special Topics in Computer Systems:**

**Cloud Computing**

**Project Proposal**

**Application Level Speculative Processing through VM Forking**

**Francesco Iorio, Alex Tessier**

With the ever increasing levels of multiprocessing and the increasing availability of cloud computing, there is frequently more processing power available than can be used by existing software applications. Efforts to multithread and parallelize the majority of existing software applications are often limited by algorithmic complexities and application data models. A large proportion of applications are event-driven, requiring either frequent user interaction or external sources of inputs to proceed. Many of these applications exhibit limited computing resources usage, and regardless of their potential for parallel execution, overall performance is confined to the shortest amount of time to respond to an input event; however it is the very event-driven nature of these applications that presents us with an opportunity. The latency between events such as user actions in an application UI can, in computing terms, be a significant amount of time that can be exploited for computation. We propose to use this time for speculative execution of application code ahead of input events.

By extending the SnowFlock[1] VM\_fork API to support speculative execution, we plan to explore the opportunistic execution of different branches in applications ahead of receiving input events, partially recovering the latency inherent in event-driven programs. By forking an entire VM based on application logic, potentially all future input events can be bundled and pre-computed on a set of local (SMP systems) or remote (clusters/cloud) computing resources before the next input event is received and then joined back to the externally facing parent VM when the received event requires committing to a specific execution branch. Computations which are no longer relevant could be discarded or, depending on application logic and event patterns, cached in case a choice validating a pre-computed result is made shortly after the initial decision. Multiple levels of speculation can be performed; the further away from the initial event group, the more likely a forked child computation is to be discarded. By using more advanced techniques such as machine learning, heuristics or even community-based statistics similar to recommender systems in case of user-facing applications [2], we believe we can improve the efficiency and utility of the system.

**Goals & Tasks**

1) set up a hardware & software environment to run vanilla snow flock on a single, multi-core system for development & testing

2) select or develop an event-driven application suitable for demonstrating speculative execution

3) design an initial framework architecture for supporting application based speculative execution

4) design and implement API on top of SnowFlock and the VMM to implement an initial version for the framework

5) test, debug to produce simple demo to show application level spec. processing in action

6) create presentation and present in class

7) generate project report

**Milestones**

I. Sept 26 - Project Proposal

II. Oct 24 - First progress report due

III. Nov 21 - Second progress report

IV. Dec 7 - Project presentation in class

V. Dec 19 - Final report due

**Timeline**

1) Complete by October 5th (1.5 w)

2) Complete by October 12th (1 w)

3) Oct 24 (1.5 w)

MILESTONE II.

4) Initial design by Nov 9th (1.5 w)

5) Initial sample implementation by Nov 21st (2.5 w)

MILESTONE III.

6) Refine demo and work on presentation to complete on Dec 7 (2.5 w)

MILESTONE IV.

7) Finish outstanding issues and complete project report by Dec 19th.

MILESTONE V.

References:

(1) [SnowFlock: Rapid Virtual Machine Cloning for Cloud Computing](http://www.cs.toronto.edu/~delara/courses/csc2231/papers/lagar.pdf)

H. Andres Lagar-Cavilla, Joseph Whitney, Adin Scannell, Philip Patchin , Stephen M. Rumble, Eyal de Lara, Michael Brudno, M. Satyanarayanan

EuroSys 2009

(2) Wei Li, Justin Matejka, Tovi Grossman, Joe Konstan & George Fitzmaurice. (2011). **Design and Evaluation of a Command Recommendation System for Software Applications.** *ACM Transactions on Computer-Human Interaction*. June 2011, Volume 18, Issue 2, Article 6. 38 pages.