This research addresses the difficult problem of verifying the results returned by potentially untrustworthy peers who have volunteered to perform computationally intensive tasks for a client in an open P2P cycle sharing system.

**CCOF (Cluster Computing on the Fly):** CCOF supports the formation of cycle sharing communities by aggregating machines on the edge of the Internet into a shared resource pool (a pauper's Beowulf cluster), providing on-demand cycles to ordinary users. CCOF assumes a dynamic, open, untrusted and insecure environment, in which each peer is potentially a host or a client (or both). This opportunistic system provides best effort service to the clients and guarantees local control for the host, including the ability to preempt guest code at any time. CCOF uses a peer-to-peer model in contrast to current client-server cycle sharing systems, such as SETI@home and the Stanford Folding project. CCOF supports application-based scheduling for four classes of applications: infinite workpile, workpile with deadlines, tree-based search, and point-of-presence applications.

**Verification through replication:** Current cycle sharing systems verify results by replicating tasks on multiple hosts followed by crosschecking of results. This method is effective under the assumptions of unlimited cycle sharing resources and no collusion among host peers because it is potentially 100% accurate. However, replication carries a fixed high overhead which may not be sustainable in a limited cycle sharing environment. We have designed an enhancement to standard replication which uses a reputation system to choose trusted hosts, and which sends out the replicas in sequence rather than in parallel. Our research results show that coupling replication with a reputation system in which hosts are rated based on past performance can significantly reduce this overhead.

**Verification through quizzes:** Verification through replication is vulnerable to attacks in which malicious hosts collude to form a majority voting block on wrong results. Furthermore, results cannot always be verified through crosschecking (e.g. unauthorized patched code for SETI@home or application-specific factors). Thus, we propose the use of quizzes – tasks with verifiable results known to the client – to test and evaluate the host peers. The quizzes are used in coordination with a reputation system, and the scheduler uses the trust values to select target hosts, thereby reducing the quiz overhead.

We are investigating two types of quizzes: stand-alone quiz and embedded quiz. With stand-alone quiz, quizzes are sent out as separate tasks which are indistinguishable from the real tasks. The CCOF scheduler on the client uses the reputation system to select a highly trusted host, the verification module uses the host's trust rating to determine the quiz sampling rate. When results are returned by the host, a wrong quiz answer causes the host to be listed on a blacklist, and results returned by that host are discarded. With embedded quiz, an obfuscation program inserts small pieces of undetectable quiz codes into the real application. The embedded quiz has much lower overhead than stand-alone quiz, but requires much more implementation effort. We are collaborating with experts from software engineering on this problem.

**Simulation Model and Preliminary Results:** We performed a simulation study of replication with reputation, and standalone quiz with reputation. We model four types of malicious nodes: the foolish malicious host always returns wrong results; the ordinary malicious host returns wrong results with a fixed probability; the smart malicious host will perform well to accumulate a good reputation before finally returning bad results with a fixed probability; the colluding malicious hosts all return the same wrong result. We modeled a dynamic system in which malicious hosts who have been blacklisted periodically rejoin the system under a new identity, and we varied the percentage of each type of malicious host. We also modeled three types of reputation systems based on the amount of knowledge they have about trusted and blacklisted hosts: local information only, sharing of information through gossiping with neighbors, and a global reputation system.

We measured accuracy, the number of correct tasks accepted by the clients divided by the total number of tasks accepted; and overhead, the number of quizzes (or replications) plus rejected tasks divided by the number of accepted tasks. Our preliminary simulation results show that quiz with reputation outperforms replication with reputation in the case of colluding cheaters. In general, the choice of verification scheme depends on the specifics of the cycle-sharing community with respect to resources (hosts), presence or absence of malicious nodes, and the standards for correctness imposed by the application itself.

Our ongoing and future work on the CCOF project includes a Wave Scheduler that exploits night-time idle cycles by timezone for workpile tasks with deadlines; algorithms for placement and scheduling of tasks for point-of-presence applications; overlay formation for cycle-sharing communities; and enforcing fairness and preventing denial of service attacks by clients who consume a disproportionate share of the resources.

For the poster, go to: http://www.cs.uoregon.edu/~liuyh/ccof/papers/sigcomm/poster.pdf

* Research Supported by NSF ANI 9977524