Exploring Self-Optimization and Self-Stabilization Properties in Biologically-inspired Cloud Computing

Jun Suzuki
Department of Computer Science
University of Massachusetts, Boston
jxs@cs.umb.edu
http://www.cs.umb.edu/~jxs/
http://dssg.cs.umb.edu/

Introduction: Cloud Computing

- Computing with Internet-accessible data centers (clouds)

- Cloud
  - is a deployment platform for network applications
    - e.g., Gmail and Google Maps
  - provides computing/networking resources (e.g., CPU cores, memory and bandwidth) on an on-demand basis for applications.
- Each app developer/provider
  - rents a part of available resources in a "pay-per-use" manner
  - runs applications with the resources.

Elastic Scaling in Clouds

- *Elastic scaling*: one of key features in clouds
- Clouds need to be
  - ***Autonomous***
    - Minimize the interruptions to/from cloud operators, app developers and app users.
  - ***Adaptive***
    - Adjusting the locations, resource allocation and availability of applications according to dynamic network conditions
      - e.g., workload and resource availability

Autonomous and Adaptive Clouds

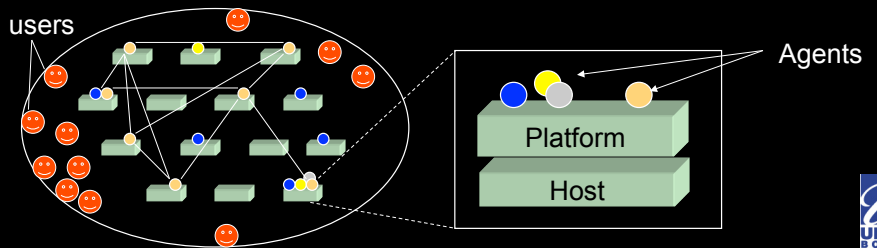
- A cloud may…
  - Co-locate multiple application components on the same host
    - Reduce response time and resource utilization cost
  - Allocate a more or less amount of resources to an app when its workload spikes or drops.
    - Match throughput, response time and resource utilization cost to a given workload
  - Replicate app components in response to high workload
    - Distribute workload over different components
  - Relocate app components to other hosts in response to low resource availability
    - Avoid potential host crashes due to resource scarcity
- ***Autonomous adaptability*** is a beneficial property for app users, app developers/providers and cloud operators.

Observation and Approach

- Observation
  - Various biological systems have already developed the mechanisms to attain autonomy and adaptability.
    - c.f. bee colonies, bird flocks, fish schools, etc.
- Approach
  - Apply biological concepts and mechanisms to design cloud applications.

The SymbioticSphere Architecture

- An application service
  - is implemented by an autonomous and distributed agent.
- A middleware platform
  - runs on a network host and operates agents.
- Each agent/platform is designed as a biological entity.
  - analogous to an individual bee in a bee colony.



Decentralization

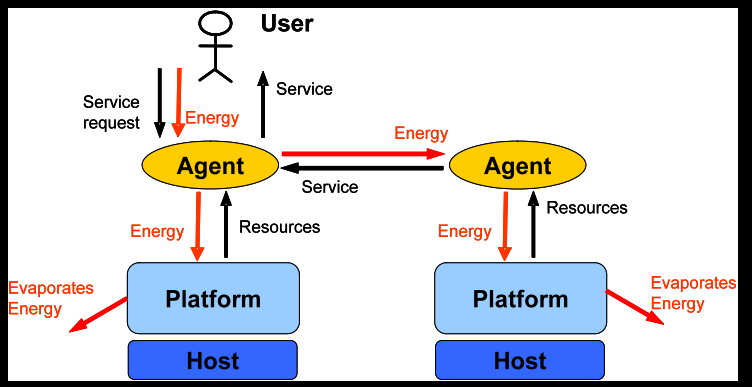
- Biological systems
  - consist of autonomous entities (e.g. bees in a bee colony)
  - no centralized entity (e.g. no leader in a bird flock)
    - Decentralization increases scalability and survivability.
- SymbioticSphere
  - biological entities → agents/platforms
    - autonomous with (biological) behaviors
      - replication, reproduction, migration, death, etc.
      - makes its own behavioral decision according to its own policy
  - no centralized entities
    - no directory servers, no resource managers

Emergence

- Biological systems
  - Useful group properties (e.g. adaptability) emerge from autonomous local interactions among group members.
- SymbioticSphere
  - Each agent/platform autonomously
    - senses local/nearby environment conditions
      - e.g. workload, resource availability
    - invokes its behavior(s) according to the current env condition
    - interacts with others.
  - Desirable system properties (e.g. adaptability) emerge from collective behaviors and interactions of agents/platforms.

Lifecycle

- Biological systems
  - Each entity strives to seek and consume food for living.
  - Some entities replicate themselves and/or reproduce offspring with partners.
- SymbioticSphere
  - Each agent stores and expends ***energy*** for living.
    - gains energy in exchange for providing its service to human users or other agents.
    - expends energy for performing its behaviors, utilizing resources (e.g. memory), and invoking another agent's service.



- Agents and platforms are modeled as different species.
- Each agent and platform
  - replicates itself or reproduces a child with a partner when its energy level becomes high.
  - dies when its energy level becomes zero.

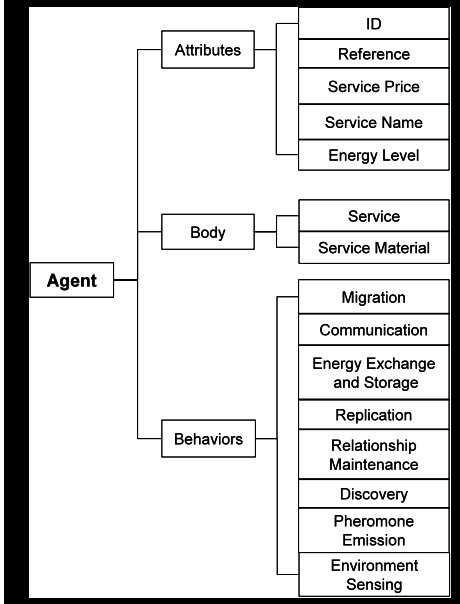
Evolution

- Biological systems
  - adjusts itself for environmental changes through generations.
- SymbioticSphere
  - Agents/platforms evolve their behavior policies (as genes) across generations.
  - Behavioral diversity among agents/platforms
    - manually by human developers
    - through mutation and crossover (in replication and reproduction)
  - Selection
    - Agents/platforms that have appropriate behavior policies will survive.
    - Agents/platforms that have inappropriate behavior policies will eventually die due to energy starvation.

Symbiosis

- Biological systems
  - Competition for food and terrain occurs regularly.
  - Several species establish mutual relationships to
    - Avoid excessive competition
    - Support with each other to survive.
- SymbioticSphere
  - Agents and platforms are selfish per se.
    - Agents may degrade platforms' adaptation, and vice versa.
  - They spontaneously cooperate to pursue their mutual benefits (i.e., gaining more energy to survive)
    - To balance and/or augment their adaptability.

Agent Structure

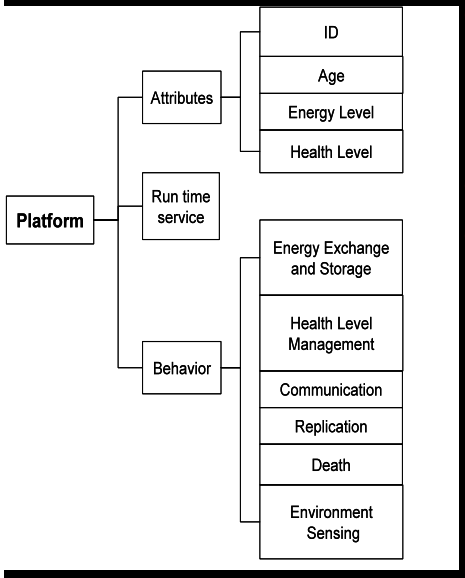


(1) Attributes (metadata)
- Energy level
- Service price (in energy unit)

(2) Body
- Functional service provided to users and other agents (e.g., web service)

(3) Behaviors
- invokes a behavior to adapt to the current environment.
  - e.g., migrate to a busier host
    - Higher chances to gain energy.

Platform Structure

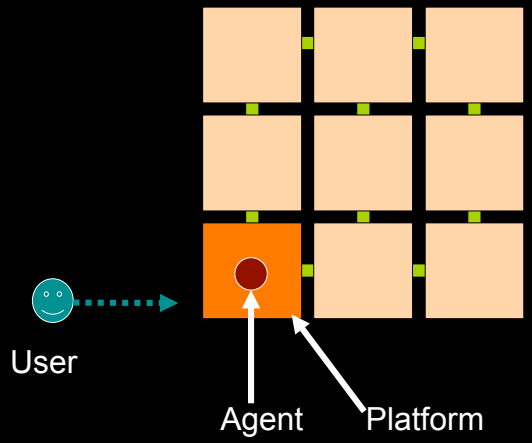


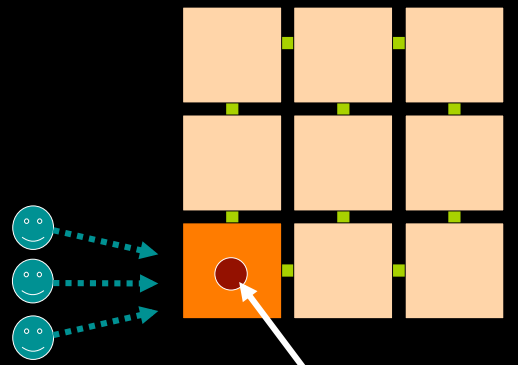
(1)Attributes
- Health level
  - a function of the age of and resource availability on a host that the platform works on.

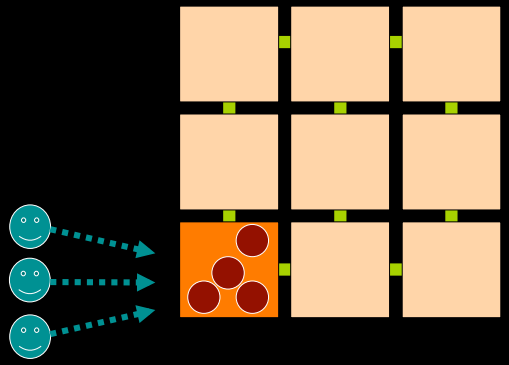
(2) Runtime Services

(3) Behaviors
- Invokes a behavior to adapt to the current environment.
  - e.g. replicates itself on a neighboring healthier host
    - Resource availability to be increased
    - A healthier host to be used for running a platform and agents

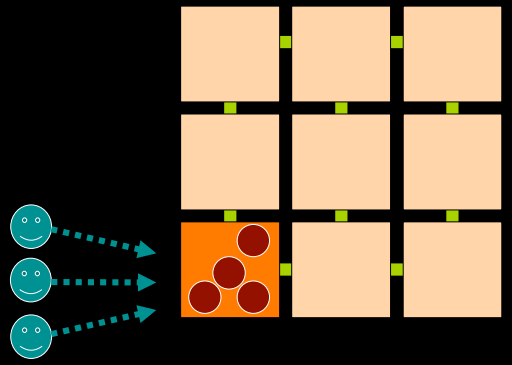
A Very High-level View of Agents' and Platforms' Behavior Invocations







Agents replicate themselves in response to increased workload.

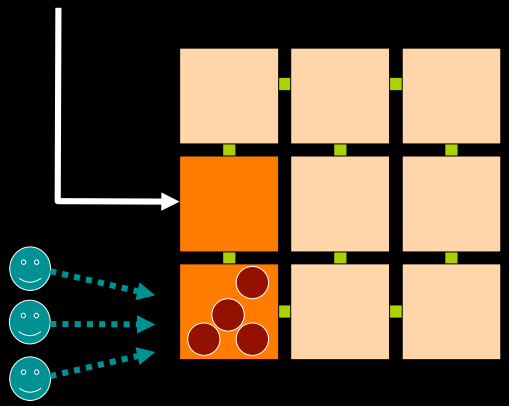


Agents replicate themselves in response to increased workload.

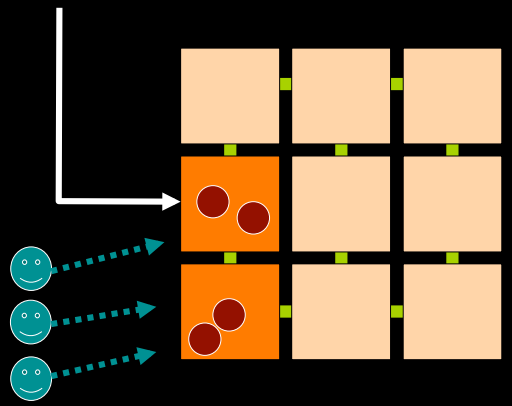
Platform's energy level goes up.

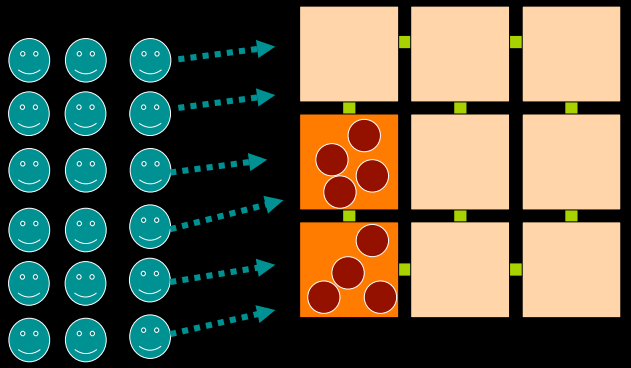
Platform's heath level (resource availability) goes down.

A platforms is replicated.



Agents migrate to balance resource utilization on network hosts.

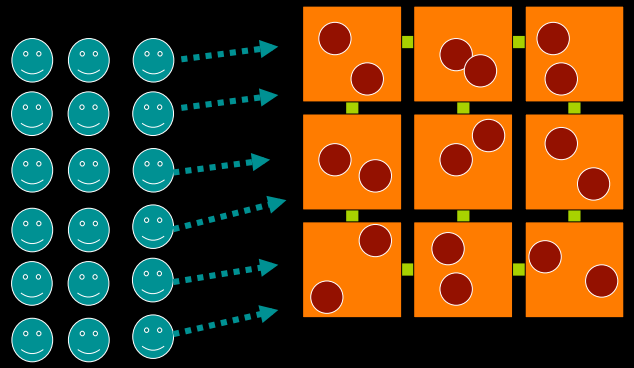




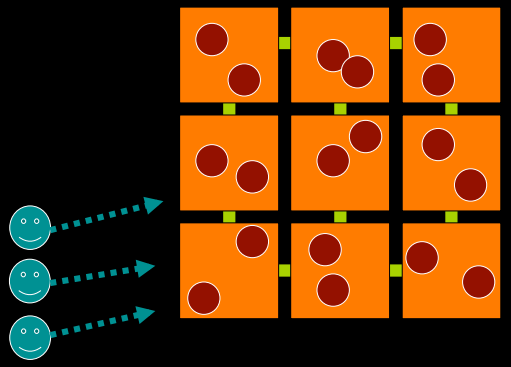
Service request rate spikes.

Agents replicate themselves.

Resource availability goes down on hosts.

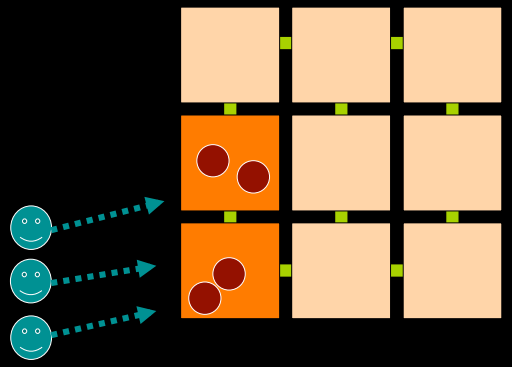


Agents and platforms spread over hosts to process incoming messages.



Service request rate drops.

Agents and platforms cannot gain enough energy (from users) to survive.



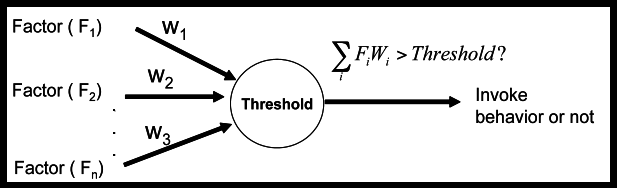
Agents and platforms die due to energy starvation.

Two Adaptation Processes in SymbioticSphere

- Self-optimization
  - Allows agents/platforms to autonomously seek their optimal behavior policies
    - With a multiobjective genetic algorithm
      - Stochastic (non-deterministic) and unstable
- Self-stabilization
  - Allows agents/platforms to autonomously seek equilibria, as symbiosis between the two species.
    - With an extensive-game between each agent and its underlying platform
      - Deterministic and stable
      - Equilibria are not always optima.

Self-Optimization Process

- Each agent/platform has its own ***policy*** for each behavior.
- A behavior policy
  - defines when to and how to invoke a particular behavior.
  - consists of *factors* (F_i), which evaluate environment conditions.
- Each factor is given a *weight* (W_i) relative to its importance.
- A behavior is invoked if the weighted sum of its factor values exceeds a threshold.

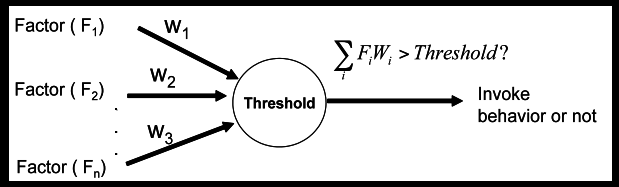


Example Behavior Policies

- Factors in agent migration behavior
  - *Service Request Ratio*:
    - (# of service requests on a neighboring platform) / (# of service requests on the local platform)
    - encourages agents to move towards users.
  - *Health Level Ratio*:
    - (health level of a neighboring host) / (health level on the local host),
    - encourages agents to move to platforms running on healthier hosts.
- Factors in platform reproduction behavior
  - *Health Level Ratio:*
    - (health level on a neighboring host) / (health level on the local host)
    - encourages platforms to replicate themselves on healthier hosts.

Behavior Policies as Genes

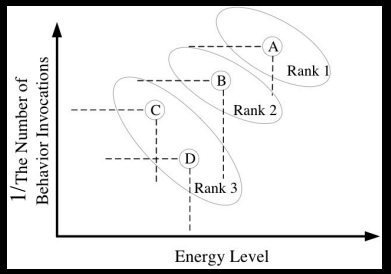
- Weights + thresholds = genes
- Each agent/platform evolve its genes (i.e., behavior policies).



Reproduction Policy			Migration Policy					Death Policy		
Wr_1^a	Wr_2^a	$\mathbf{Tr^a}$	Wm_1^a	Wm_2^a	Wm_3^a	Wm_4^a	$\mathbf{Tm^a}$	Wd_1^a	Wd_2^a	$\mathbf{Td^a}$

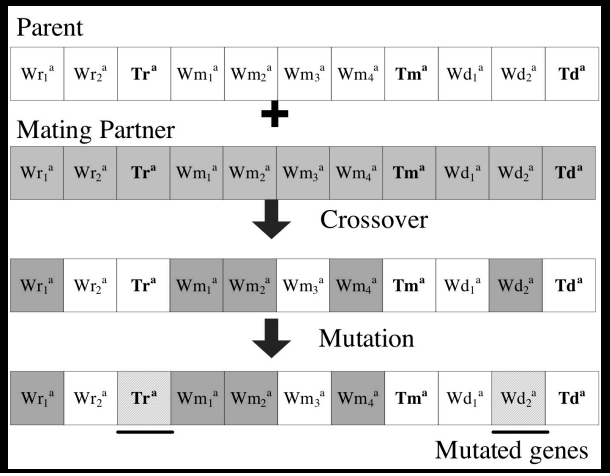
Reproduction: Mate Selection

- When an agent/platform invokes the reproduction behavior, it finds a mate to reproduce offspring.
- Each agent/platform ranks other agents/platforms running on the local and neighboring hosts with respect to
  - Energy level (to be maximized)
  - # of behavior invocations (to be minimized)



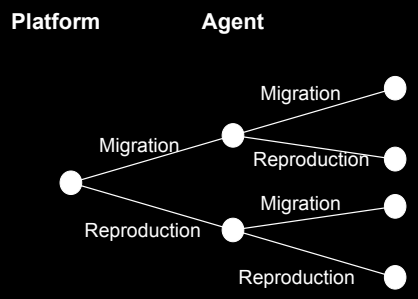
Reproduction: Crossover and Mutation

- When an agent/platform selects the best-ranked one as a mate.
  - If it is the best-ranked one, it replicates itself.



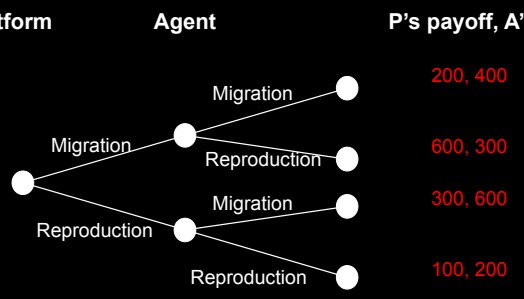
Self-Stabilization Process

- Agent-initiated symbiosis
  - A sequence of an agent's behavior invocation and its underlying platform's behavior invocation
- Platform-initiated symbiosis
  - A sequence of a platform's behavior invocation and its local agent's behavior invocation



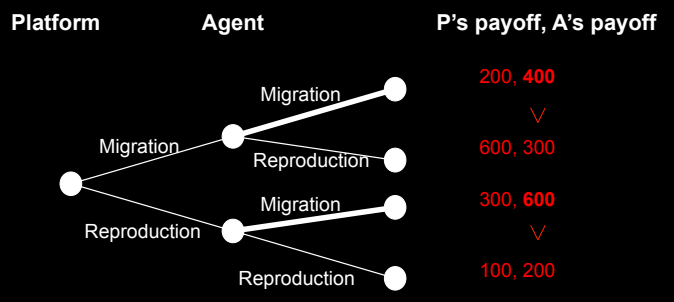
Payoff Computation

- Compute a payoff for each behavior sequence
  - Based on estimated environment conditions
    - Platform's health level
    - # of agents per platform
    - Service request rate
- Positive/negative payoff → environment conditions are expected to improve/degrade.



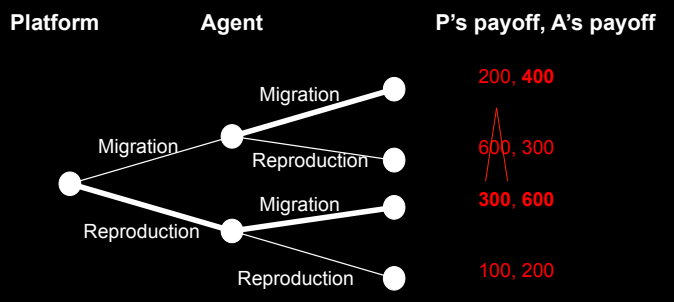
Finding an Equilibrium

- An extensive game always has at least one Nash equilibrium.
- Backward induction can determine equilibria.



Finding an Equilibrium

- The sequence of *reproduction-migration* is an equilibrium.
  - Both players cannot gain higher payoffs by changing their behaviors.
  - Represents a platform-agent symbiosis that the two species can agree upon.

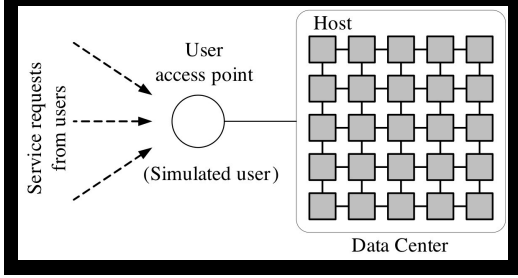


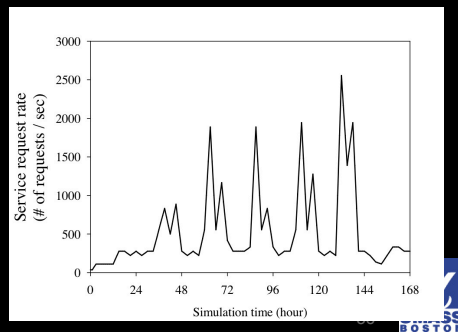
Integration of Self-Optimization and Self-Stabilization

- In each generation…
  - For each agent…
    - Invoke an agent-initiated symbiotic sequence of behaviors
      - If both agent payoff and platform payoff are positive
      - Otherwise, invoke the agent's behavior with its behavior policy
  - For each platform…
    - Invoke a platform-initiated symbiotic sequence of behaviors
      - If both agent payoff and platform payoff are positive
      - Otherwise, invoke the platform's behavior with its behavior policy

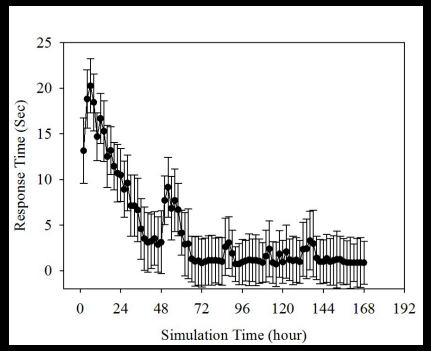
Simulation Evaluation

- 25 simulated hosts
- Workload trace of the 1998 Soccer World Cup's official web site
- One agent and one platform are initially deployed on each host
  - Initially, agents and platforms have randomly-generated behavior policies.

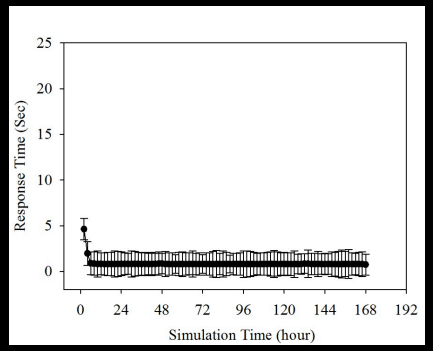




Response Time

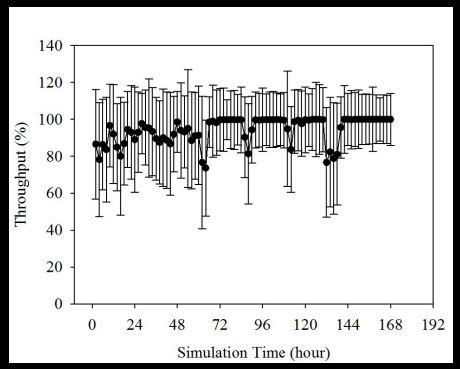


w/ Self-Optimization Process

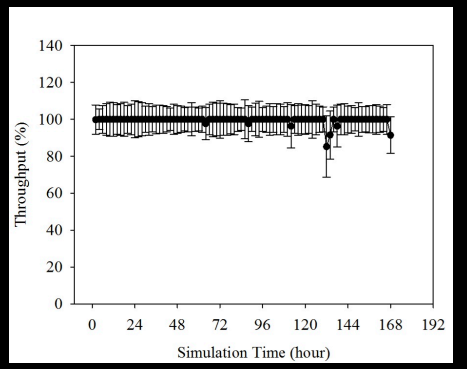


w/ Self-Optimization & Self-Stabilization Processes

Throughput



w/ Self-Optimization Process



w/ Self-Optimization & Self-Stabilization Processes

Conclusions

- Proposed and evaluated a bio-inspired architecture to build self-optimizable and self-stabilizable cloud applications
- On-going work
  - Empirical evaluation with 25 hosts
    - Empirical evaluation results are qualitatively similar to simulation results.
  - Larger-scale simulation evaluation with 400 hosts
- Future directions
  - Security via adaptation
  - Better integration of self-optimization and self-stabilization processes
  - Empirical evaluation with “real” clouds
    - Amazon EC2, Eucalyptus, etc.