

Robust Resource Allocation in a Client/Server Hybrid Network for Virtual World Environments

Luis D. Briceño¹, H. J. Siegel^{1,2}, Anthony A. Maciejewski¹, Ye Hong¹, Brad Lock¹,
 Mohammad Nayeem Teli², Fadi Wedyan², Chuck Panaccione², and Chen Zhang²

¹Electrical & Computer Engineering, ²Department of Computer Science

{ldbricen, hj, aam}@colostate.edu

This research was supported by the NSF under grant CNS-0615170 and by the Colorado State University George T. Abell Endowment

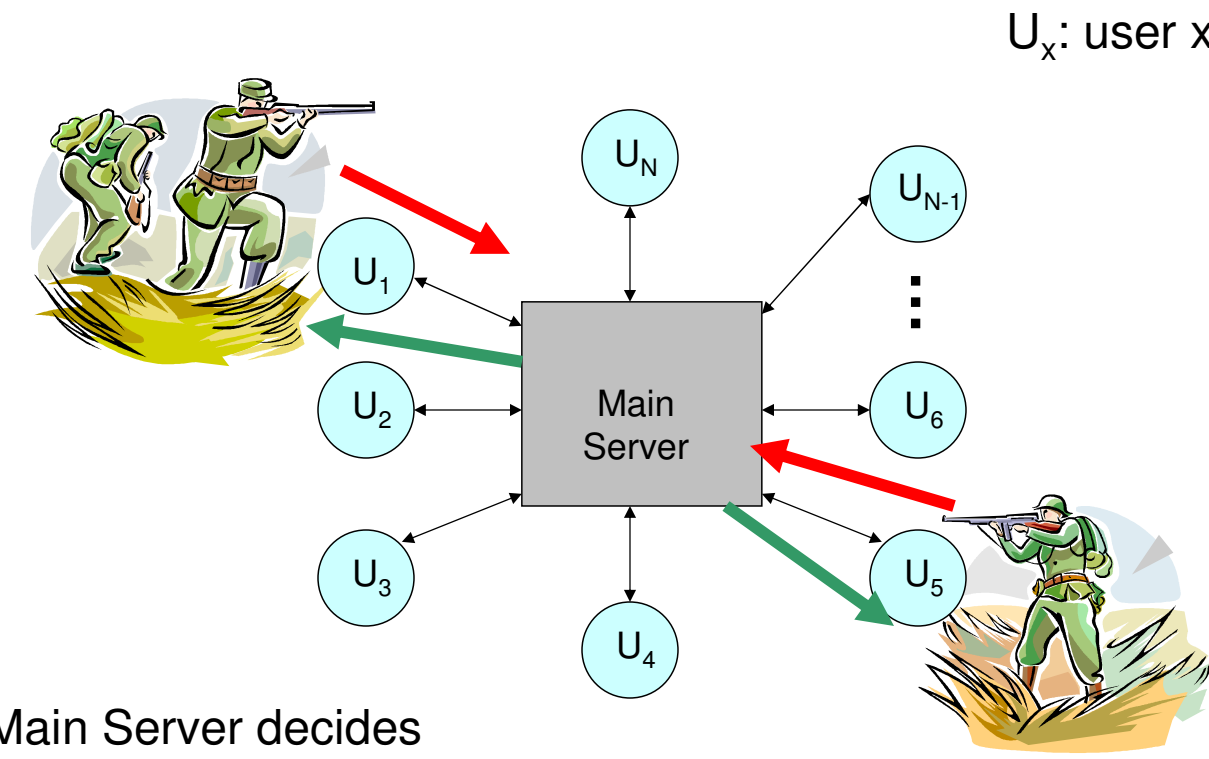
Introduction

- What is heterogeneous computing (HC)?
 - HC is the coordinated use of different types of machines, networks, and interfaces to maximize performance
- What is a virtual world environment?
 - a virtual world environment is a multi-user program that creates a game world
- What is the goal of this research?
 - in this study we maximize the robustness in the system
 - robustness for this study is the number of additional players that can be added

Cost in an MMOG

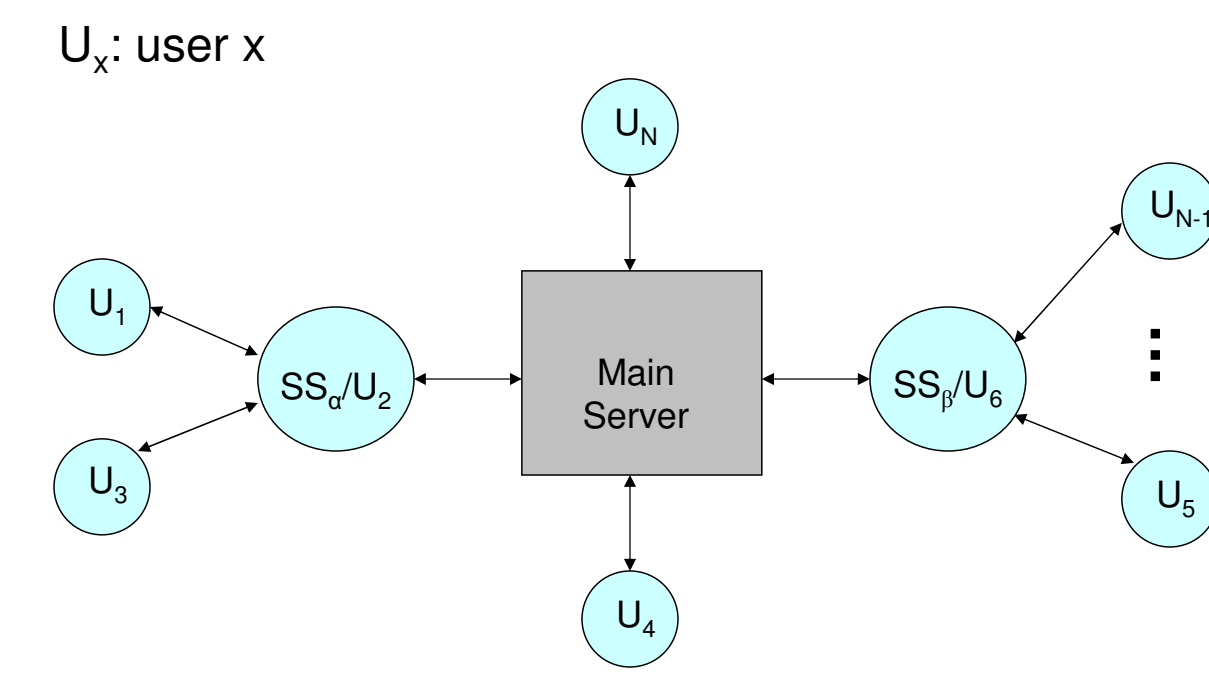
- support of Massive Multiplayer Online Games (MMOGs) has a significant cost:
 - game server configuration costs
 - customer service
 - technical problems during and after launch
- many MMOGs use a logical single server architecture
 - World of Warcraft game world is divided into shards that are processed by a single server
- disadvantages of a logical single server architecture:
 - initial procurement of servers is expensive
 - hard to scale architecture based on demand

Example of Interaction in an MMOG



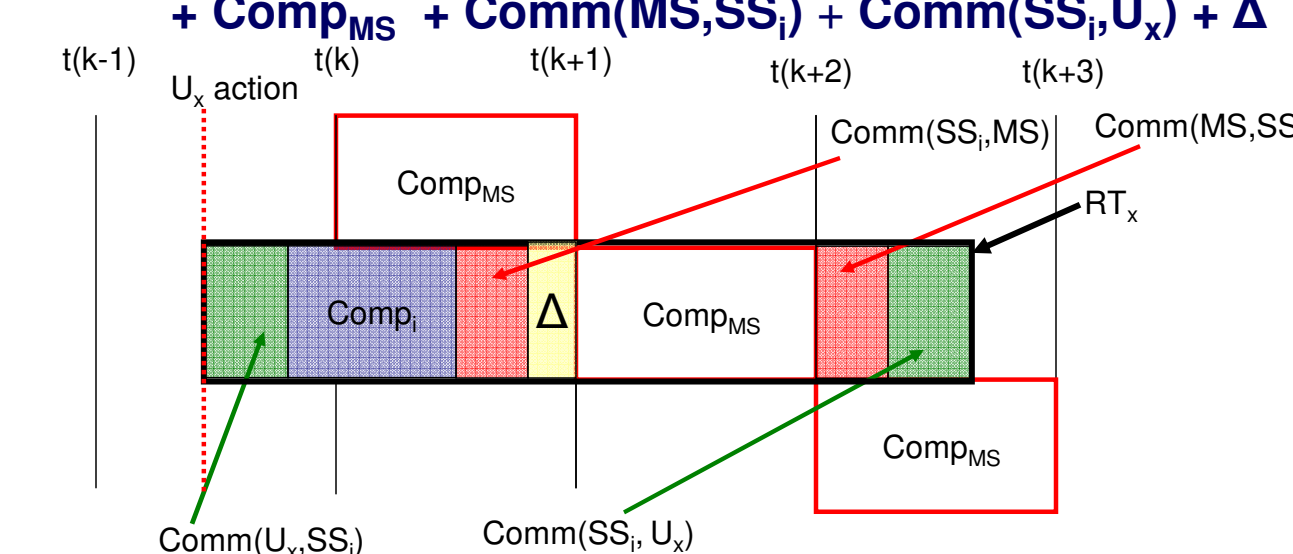
Proposed Solution

- users' computers are converted into secondary servers (SS)
- SS are used to assist the Main Server in computation



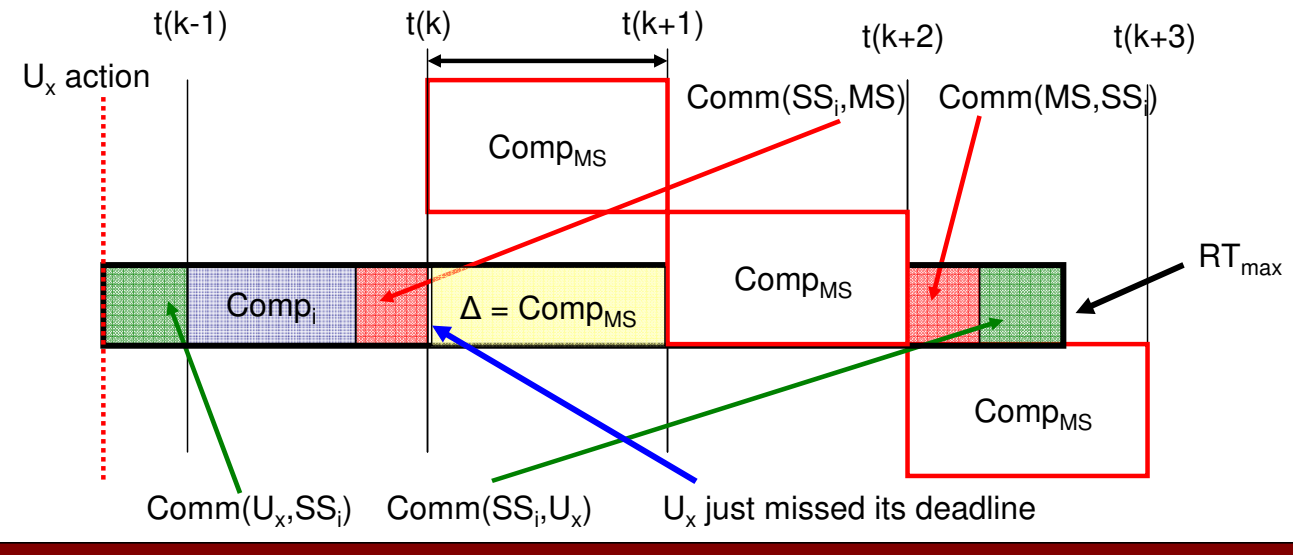
Response Time for User X (RT_x)

RT_x : response time from user x (U_x) to the server and back
MS: Main Server; **SS**: secondary server
Comp_C: computation at node C
Comm(A,B): communication time from A to B
 Δ : uncertainty in the arrival time of the user action to the MS

$$RT_x = \text{Comm}(U_x, SS) + \text{Comp}_x + \text{Comm}(SS, MS) + \text{Comp}_{MS} + \text{Comm}(MS, SS) + \text{Comm}(SS, U_x) + \Delta$$


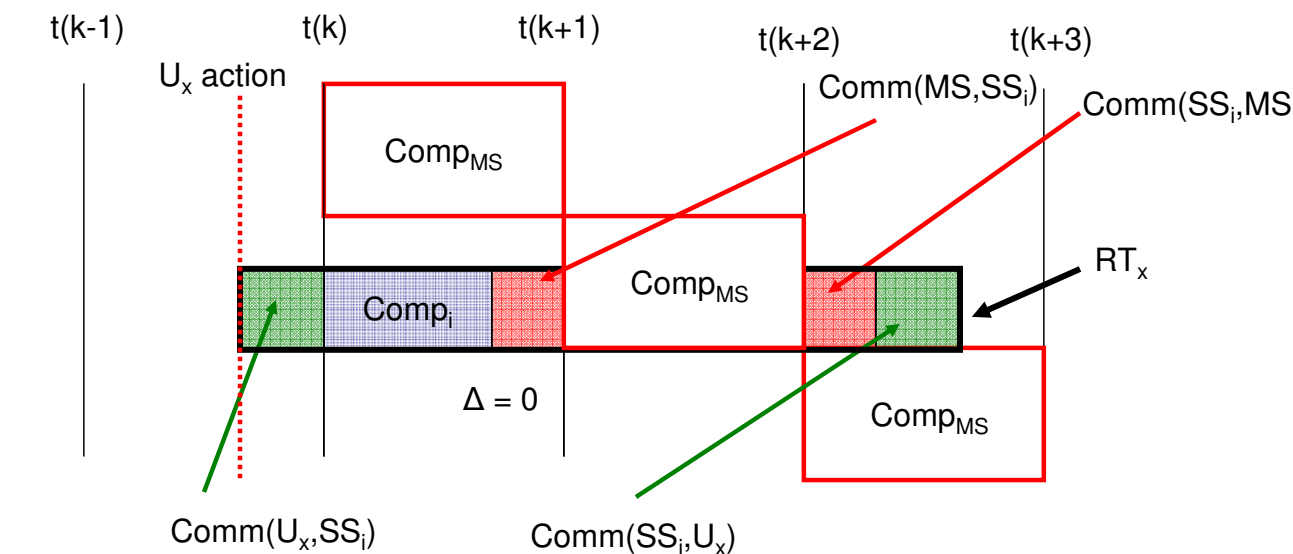
Maximum Response Time (RT_{max})

- user x missed the $t(k)$ deadline by $(\text{Comp}_{MS} - \Delta)$ time units (with $0 < \Delta < \text{Comp}_{MS}$)
- the worst time for U_x to interact with the environment is when $\Delta \rightarrow \text{Comp}_{MS}$
- the worst time for any user to interact with the environment can be bounded by $RT_{max} = \max_{\forall U_x} (RT_x)$ with $\Delta = \text{Comp}_{MS}$



Minimum Response Time (RT_{min})

- assume $RT_x \geq RT_{min}$ (for any U_x)
- minimum time any user requires to interact with the virtual world environment is RT_{min} time units
- $RT_{min} = \min_{\forall U_x} (RT_x)$ with $\Delta = 0$



Robustness Metric

- robustness of the system
 - based on perceived fairness
 - how responsive does the system need to be to avoid users from realizing the system is unfair
- a constraint of is used to limit the worst responsiveness the system allows (β_{max})
- Δ_{max} is the time window (value is fixed)
- 2 constraints should be used
 - $RT_{max} \leq \beta_{max}$
 - $RT_{min} \geq (RT_{max} - \Delta_{max})$

Three Robustness Questions

- **what behavior of the system makes it robust?**
 - the system is robust if $RT_{max} \leq \beta_{max}$ and $RT_{min} \geq (RT_{max} - \Delta_{max})$
- **what uncertainties is the system robust against?**
 - the system is robust against new players joining the game
- **quantitatively, exactly how robust is the system?**
 - the number of new players that can be connected directly to the Main Server and still have a system that meets the robustness criteria

Heuristics

- **ROAR**: greedy heuristic that combines exploration
- **Tabu Search**: combines global and local search
- **Discrete Particle Swarm Optimization**: mimics flocking behavior of birds to explore solutions
- **Genetic Algorithm**: mimics evolution by combining solutions to produce better solutions (or offspring)
- **Ant Colony Optimization**: mimics the behavior of ants to produce solutions
- **Generic Players Iterative Maximization**: combines random search with a greedy search

heuristics are used to maximize the number of additional players that can be connected to the Main Server while maintaining the desired performance metrics

Pending Work

- The implementation of the heuristics is done
 - more experimentation with the parameters of these heuristics is required