Ludwig-Maximilians-Universität München
Institut für Informatik
Prof. Dr. Matthias Schubert
Felix Borutta

Munich, May 2, 2019

Managing Massive Multiplayer Online Games

SS 2019

Exercise Sheet 1: Modelling of Time

The assignments are due May 8, 2019

Assignment 1-1 *Time*

Consider an abstract game where the goal of two players $\{S_1, S_2\}$ is to collect as many coins $\{m_1, ..., m_9\}$ as possible. Each coin can only be collected once, if two players want to collect the same coin at the same time, the coin is divided and the score of both players increases by half a coin. Winner is who has most coins eventually.

We assume a client-server architecture where a player S_i sends an action request of the form $Collect(m_j, t, t')$ via his client to the server. m_j is the ID of the coin that player S_i wants to collect, t represents the time when the client sent his request and t' is the time when the request arrives at the server.

Consider an instance of this game in which the following action requests are sent.

Player S_1		
Coin	Time (Client)	Time (Server)
m_1	1	4
m_2	2	3
m_3	3	4
m_4	4	6
m_5	5	8
m_6	6	7
m_7	7	8
m_8	8	10
m_9	9	11

Player S_2		
Coin	Time (Client)	Time (Server)
m_1	1	8
m_4	1	7
m_3	2	10
m_2	2	9
m_8	2	9
m_7	3	10
m_9	3	11
m_5	4	10
m_6	4	13

How many coins do the players have at the end of the game when using the following models of time:

(a) Turn-based (alternately): Before a player can perform an action she has to wait until the other player has finished her action. Player S_1 begins. If a player wants to pick up a coin which does not exist any more it is her move again.

(b) Turn-based (simultaneously): In each round pairs of players' actions are performed simultaneously. Players have to wait for the other players' actions.

(c) Real-time (Server-sided): Action requests are executed immediately when they arrive at the server. If a coin should be picked up which does not exist any more, the request is discarded.

(d) Real-time (Client-sided): Action requests are collected by the server and then executed in the order in which they were sent by the clients. If a coin should be picked up which does not exist any more, the request is discarded.

(a)

turn	1	2	3	4	5	6	7	8	9	10	Σ
S_1	m_1	×	m_2	×	m_5	×	m_6	×	m_9	×	5
S_2	×	m_4	×	m_3	×	m_8	×	m_7	×	×	4

(b)

turn	1	2	3	4	5	6	7	8	9	Σ
S_1	$m_1/2$	m_2	$m_3/2$	×	m_5	m_6	×	×	×	4
S_2	$m_1/2$	m_4	$m_3/2$	×	m_8	m_7	m_9	×	×	5

(c)

t	1	2	3	4	5	6	7	8	9	10	11	12	13	Σ
S_1	×	×	m_2	m_1, m_3	×	m_4	m_6	m_5, m_7	×	×	$m_9/2$	×	×	7.5
S_2	×	×	×	×	×	×	×	×	m_8	×	$m_9/2$	×	×	1.5

(d)

t	1	2	3	4	5	6	7	8	9	Σ
S_1	$m_1/2$	$m_2/2$	×	×	×	×	×	×	×	1
S_2	$m_1/2, m_4$	$m_2/2, m_3, m_8$	m_7, m_9	m_5, m_6	×	×	×	×	×	8

Please note the discrepancy of the final results for the real-time models. Though player S_2 might be the better player (regarding the timestamps when the actions are performed at the client), she suffers from the server-sided model due to slow request transmissions.

Discuss the advantages and disadvantages of these models!

Turn-based:

Advantages:

- Easy to manage (e.g., no concurrency and conflict management)
- Fair in the sense that for instance hardware discrepancies, resp. network latency do not take effect

Disadvantages:

- player attendance is needed
- gameplay may allow for simultaneous player actions (might be fixed with simultaneous turns (as in (b)), which in turn would require conflict management)

Real-time (requests are processed immediately when they arrive at the server):

Advantages:

- can be based on standard solutions (e.g., DBS)
- allows concurrency: reduces waiting time for other players

Disadvantages:

- no synchronization between game and real time, thus: no control of max. amount of actions per time unit (e.g., slow hardware, network may be disadvantaged)
- Simultaneous actions are impossible (serialization) (example: healing vs. suffering damage at the same time, cf. slide 20)

Tick-Systems / Soft-Real-Time Simulation (a couple of actions are accumulated and processed as bunch):

Advantages:

- synchronizes game-time and real-time
- fair rules for actions per time-unit
- concurrency (no serialization of actions)
- fast players do not get disadvantaged

Disadvantages:

- potential lags (server does not finish computing a tick in time) require lag handling strategies
- Conflict resolution for contradictory actions
- chronological order (all actions generated within one tick are considered concurrent), hence conflict resolution for concurrent actions is necessary

Hopefully I got everything from our discussions. ;-)