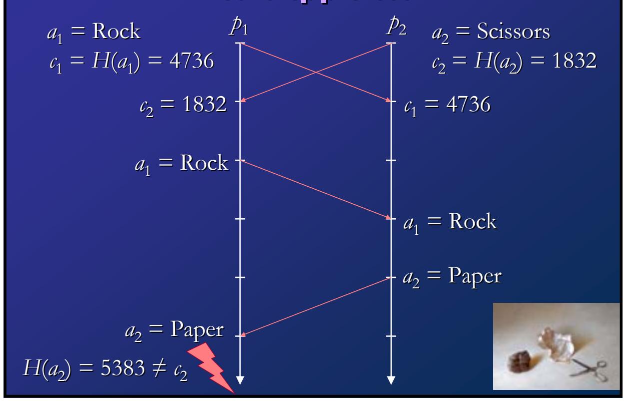
Lockstep protocol

1. Announce a commitment to an action.
   - commitment can be easily calculated from the action but the action cannot be inferred from the commitment
   - formed with a one-way function (e.g., hash)
2. When everybody has announced their commitments for the turn, announce the action.
   - everybody knows what everybody else has promised to do
3. Verify that the actions correspond to the commitments.
   - if not, then somebody is cheating…

Lockstep protocol



Loosening the synchronization 1(2)

- the slowest player dictates the speed
  - short turns
  - time limits for the announcements
- asynchronous lockstep protocol
  - sphere of influence: synchronization is needed only when the players can affect each other in the next turn(s)
  - otherwise, the players can proceed asynchronously

Loosening the synchronization 2(2)

- pipelined lockstep protocol
  - player can send several commitments which are pipelined
  - drawback: look-ahead cheating if a player announces action earlier than required
- adaptive pipeline protocol
  - measure the actual latencies between the players
  - grow or shrink the pipeline size accordingly

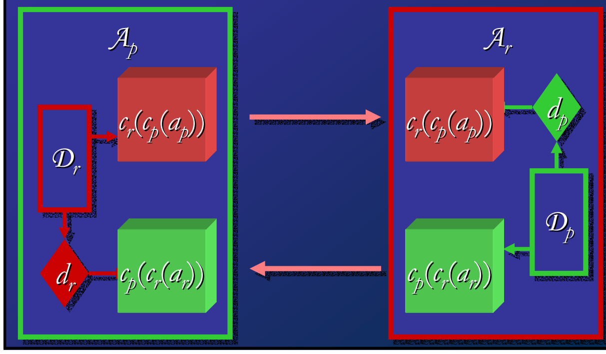
Drawbacks of the lockstep protocol

- requires two separate message transmissions
  - commitment and action are sent separately
  - slows down the communication
- requires a synchronization step
  - the slowest player dictates the pace
    - improvements: asynchronous lockstep, pipelined lockstep, adaptive pipeline lockstep
- does not solve the inconsistency problem!

Idea #1: Let's get rid of the repeat!

- send only a single message
  - but how can we be sure that the opponent cannot learn the action before annoucing its own action?
- the message is an active object, a *delegate*
  - program code to be run by the receiver (host)
  - delegate's behaviour cannot be worked out by analytical methods alone
  - guarantees the message exchange on a possibly hostile environment
- delegate provides the action once the host has sent its own action *using* the delegate

Example with two players

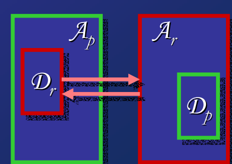


Threats

- what if the host delays or prevents the delegate's message from getting to its originator?
  - the host will not receive the next delegate until the message is sent
- what if the originator is malicious and the delegate spies or wastes the host's resources?
  - sandbox: the host restricts the resources available to the delegate
- how can the delegate be sure that it is sending messages to its originator?
  - communication check-up

Communication check-up

- the delegate sends a unique identification to its originator
  - static and dynamic information
- the delegate waits until the originator has responded correctly
- check-ups are done randomly
  - probability can be quite low
  - host cannot know whether the transmission is the actual message or just a check-up



Idea #2: Peer pressure

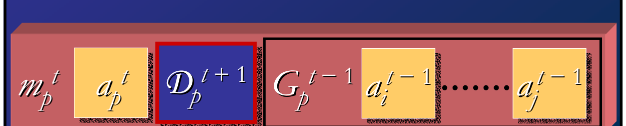
- players gossip the other players' actions from the previous turn(s)
- compare gossip and recorded actions; if there are inconsistencies, ban the player
  - cheating is detected only afterwards
  - gossiping imposes a threat of getting caught
- gossip is piggybacked in the ordinary messages
  - no extra transmissions are required
- how to be sure that the gossip is not forged?
  - rechecking with randomly selected players

How much is enough?

- example: 10 players, 60 turns, 1 cheater who forges 10% of messages, gossip from one previous turn
  - 1% gossip: *P*(cheater gets caught) = 0.44
  - 5% gossip: *P*(cheater gets caught) = 0.91
  - 10% gossip: *P*(cheater gets caught) = 0.98
- example: 100 players, 60 turns, 1 cheater who forges 10% of messages
  - 1% gossip: *P*(cheater gets caught) = 0.98
- example: 10 players, 360 turns, 1 cheater who forges 10% of messages
  - 1% gossip: *P*(cheater gets caught) = 0.97

Message

- action for the current turn t
- delegate for the next turn $t + 1$
- set of actions (i.e., gossip) from the previous turn $t - 1$



Collusion

- imperfect information games
  - infer the hidden information
  - outwit the opponents
- collusion = two or more players play together without informing the other participants
- how to detect collusion in online game?
  - players can communicate through other media
  - one player can have several avatars

Analysing collusion

- tracking
  - determine who the players are
  - but physical identity does not reflect who is actually playing the game
- styling
  - analyse how the players play the game
  - requires a sufficient amount of game data
  - collusion can be detected only afterwards

→ no pre-emptive nor real-time counter-measures

Collusion types

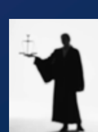
- active collusion
  - cheaters play more aggressively than they normally would
  - can be detected with styling
- passive collusion
  - cheaters play more cautiously than they normally would
  - practically undetectable

Offending other players

- acting against the 'spirit' of the game
  - problematic: is camping in a first-person shooter cheating or just a good tactic?
  - some rules are 'gentlemen's agreements'
- examples
  - killing and stealing from inexperienced and ill-equipped players
  - gangs and ghettoization of the game world
  - blocking exits, interfering fights, verbal abuse

Upholding justice

- players handle the policing themselves
  - theory: players take the law into their own hands (e.g., militia)
  - reality: gangs shall inherit the game world
- systems records misconducts and brands offenders as criminals
  - theory: bounties and penalties prevent crimes
  - reality: throw-away avatars commit the crimes
- players decide whether they can offend/be offended
  - theory: players know what kind of game world they want
  - reality: how to offend you? let me count the ways…



Recapitulation: Outline of the course

8. Communication layers
   - physical platform
   - logical platform
   - networked application
9. Compensating resourse limitations
   - aspects of compensation
   - protocol optimization
   - dead reckoning
   - local perception filters
   - synchronized simulation
   - area-of-interest filtering
10. Cheating prevention
    - technical exploitations
    - rule violations

Examinations 1 (2)

- examination dates
  1. January 16, 2006
  2. February 13, 2006
  3. March 2, 2006
- check the exact times and places at `http://www.it.utu.fi/opetus/tentit/`
- if you are *not* a student of University of Turku, you must register to receive the credits
  - further instructions are available at `http://http://www.tucs.fi/education/courses/participating_courses.php`

Examinations 2 (2)

- questions
  - based on both lectures and lecture notes
  - two questions, à 5 points
  - to pass the examination, at least 5 points (50%) are required
  - grade: $g = \lceil p - 5 \rceil$
  - questions are in English, but you can answer in English or in Finnish
- remember to enrol in time!