ARIVU: Power-Aware Middleware for Multiplayer Mobile Games

Anand Bhojan
COM2-04-26, School of Computing
www.comp.nus.edu.sg/~bhojan
banand@comp.nus.edu.sg ph: 651-67351

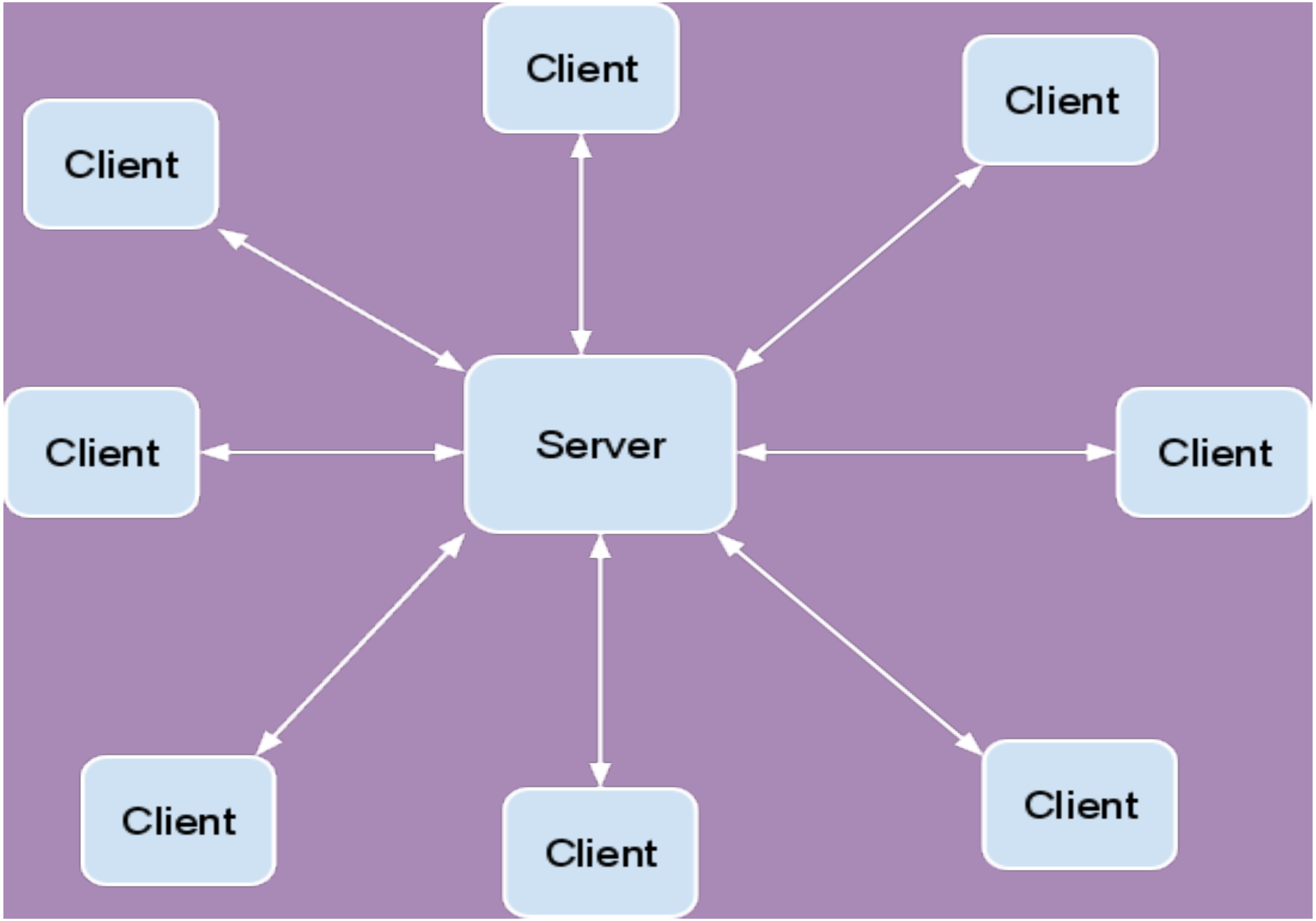
+ References

- **Bhojan Anand**, Akhihebbal L. Ananda, Mun Choon Chan and Rajesh Krishna Balan, "ARIVU: Making Networked Mobile Games Green - A Scalable Power-Aware Middleware", MOBILE NETWORKS AND APPLICATIONS, Springer Netherlands,(DOI: 10.1007/s11036-011-0312-8, URL:http://dx.doi.org/10.1007/s11036-011-0312-8), Feb 2012.

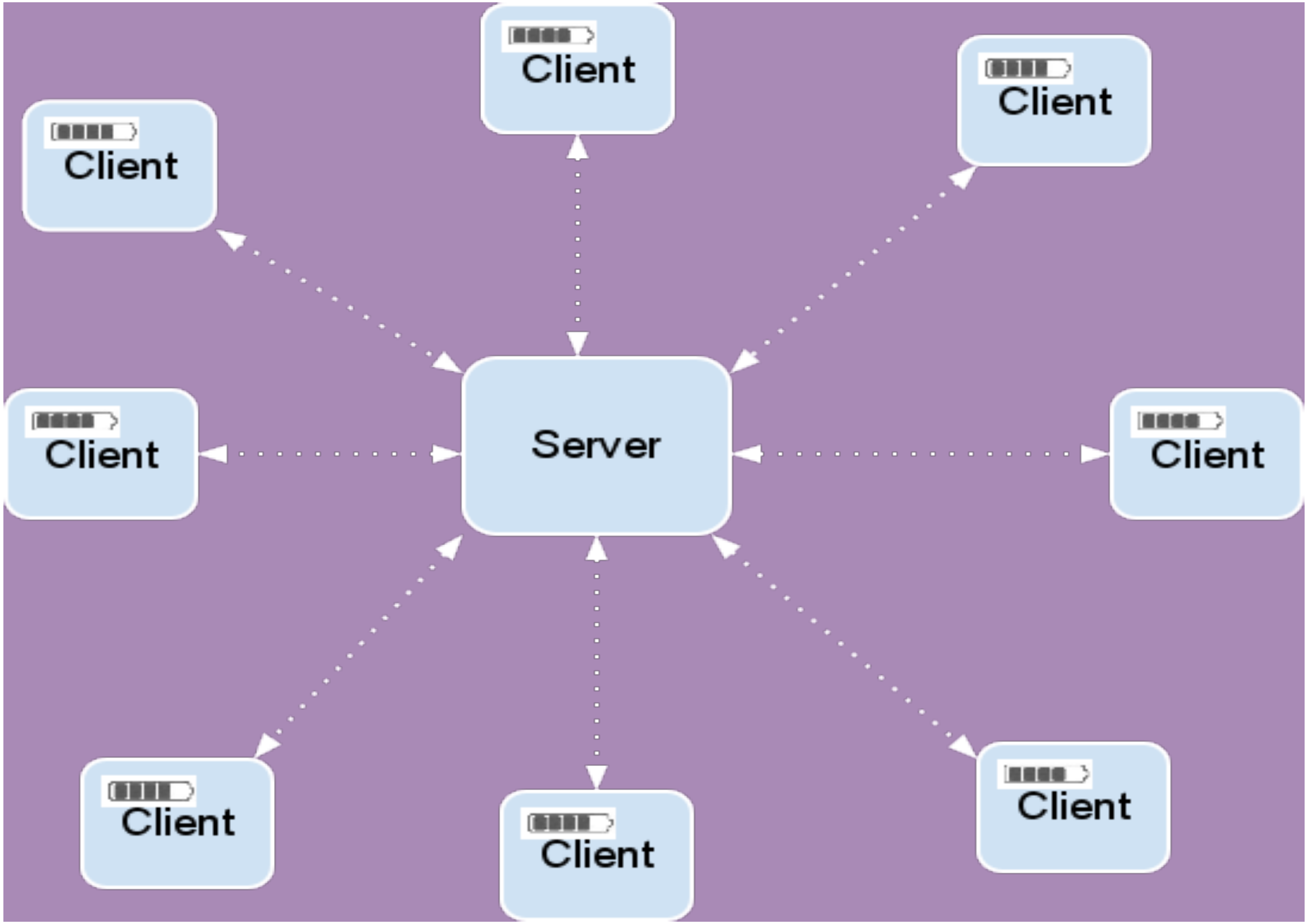
- K Thirugnanam, **Bhojan Anand**, J Sebastian, PG Kannan, AL Ananda, RK Balan, and MC Chan, "Dynamic Lookahead Mechanism for Conserving Power in Multi-Player Mobile Games," IEEE INFOCOM 2012.

- **Bhojan Anand**, Karthik Thirugnanam, Jeena Sebastien, Pravein Govindan Kannan, Akhihebbal L. Ananda, Mun Choon Chan and Rajesh Krishna Balan, "*Adaptive Display* ***Power*** *Management for* ***Mobile Games***", Proceeedings of ACM MobiSys 2011

Multiplayer Game



Mobile Multiplayer Game



Complex User Demand

Data rate,
Computational
power

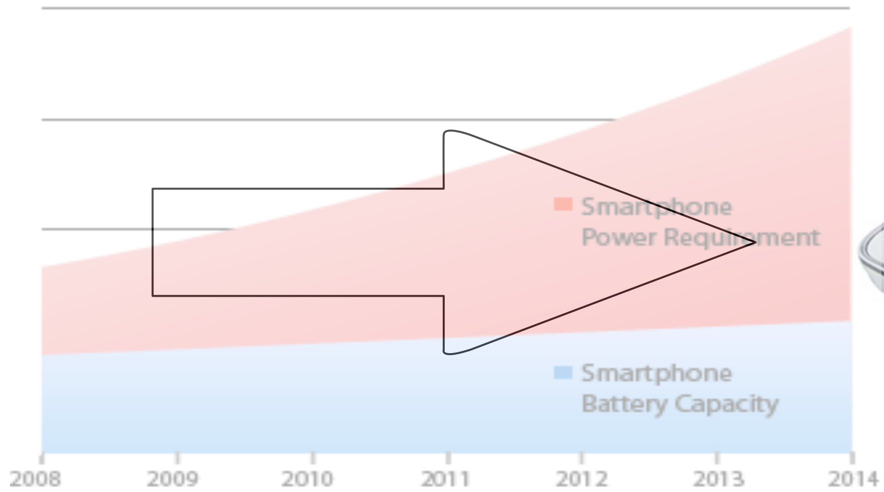
QoS (user experience)
Energy
efficiency

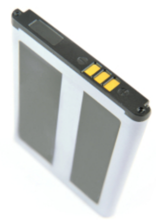
User demand is shifting

Simple to complex requirement

Need for Power Management

Demand and Supply





Widening gap between "Demand and Supply".

"Improvements in battery technology, while steady, no longer happen at the breakneck speed of younger technology like smartphones," says Keith Nowak of phone and tablet maker HTC.

Need for Power Management

Usability Perspective

Competitive Advantage!

Platform (Phone)	Type	Battery Type Battery Type	GSM talk, standby	WCDMA talk, standby	Quake III play
Symbian (Nokia X7 [52])	OIA	1200 mAh Li-Ion (BL-5K)	6h 30min, 450h	4h 30min, 450 h	- -
Apple iOS (iPhone 4 [7])	VIA	1420 mAh Li-Po	14h, 300h	7h, 300h	- -
Android (Google Nexus S [33])	OIA	1500 mAh Li-Ion 1500 mAh Li-Ion	14h, 713h	6h 42 min, 427h	- -
Android (HTC Desire HD [37])	OIA	1230 mAh Li-Ion 1230 mAh Li-Ion	8h 10 min,420h	5h 20 min, 490h	*1h 50min

* - To estimate this, we played Quake III Arena on a HTC Desire HD smartphone, for 5 minutes, and determined the percentage of battery power drained.

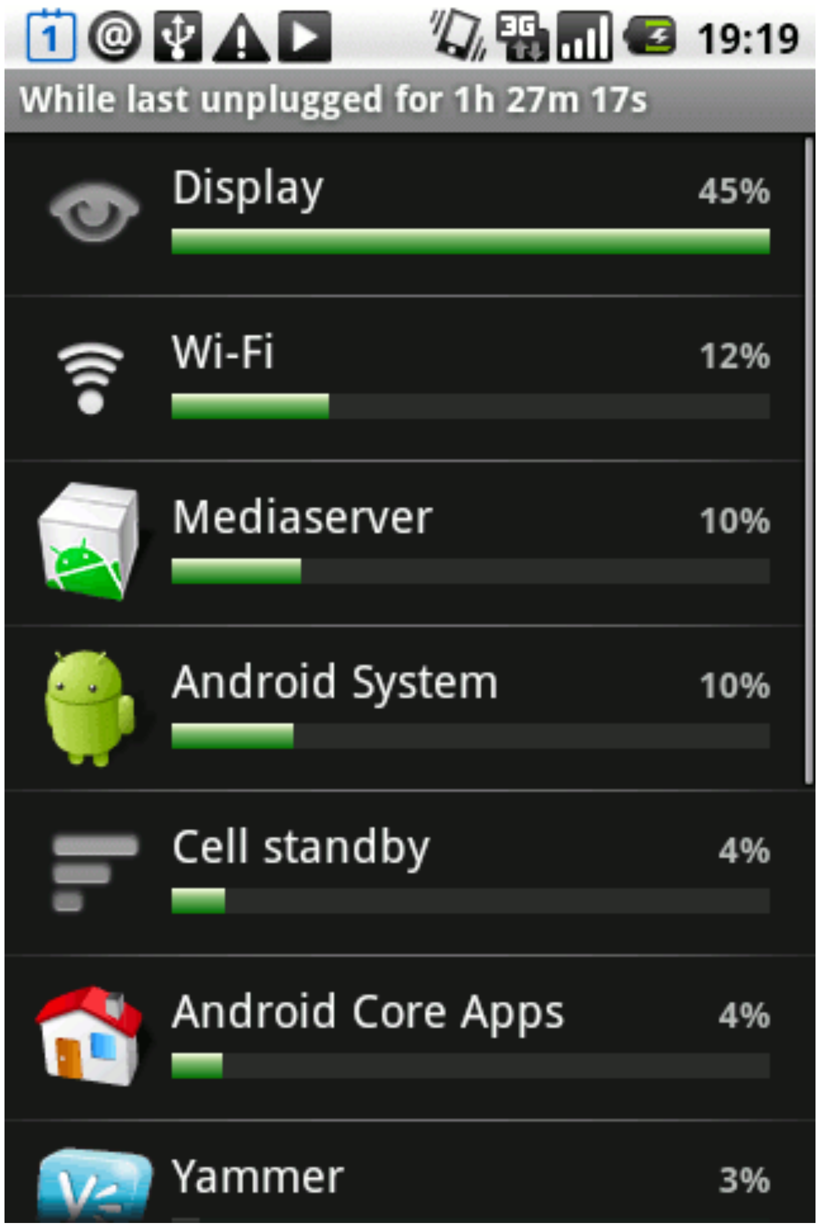
Mobile game is one of the most rapidly growing areas in today's consumer technology. Games alone account for more than 50% of current iPhone application downloads.

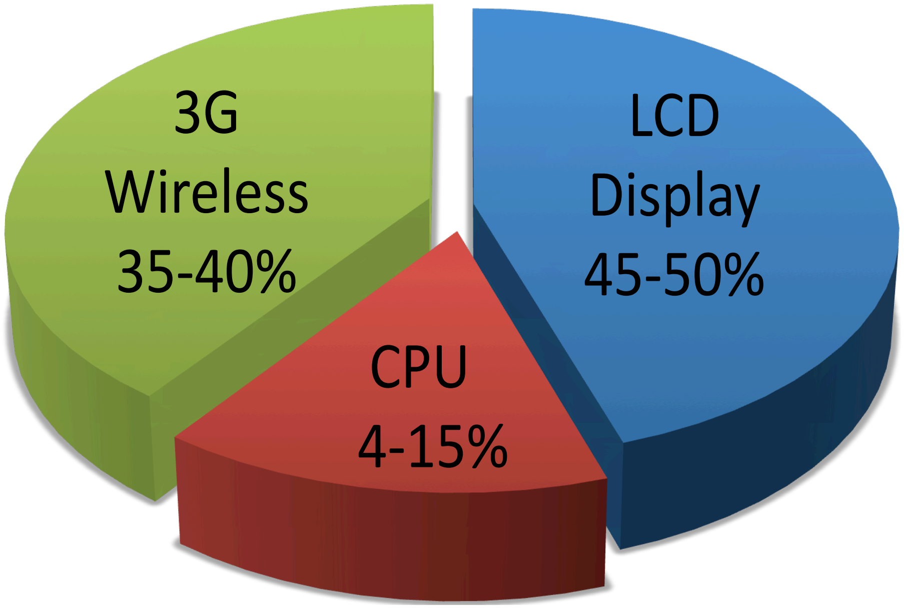
Need for Power Management

Environmental and Financial Perspective

- Mobile devices - Growing as one of the major contributor for ICT energy consumption.
  - Mobile phone subscriptions worldwide has surpassed 5 billion
  - Annual electricity consumption for a mobile phone 11 KWh per year.
  - Mobile devices: 12% of total ICT energy. which is (55 million MWh / 452.3 million MWh per year)
- Contribute to Green House Gases and Energy Budget

Energy Distribution





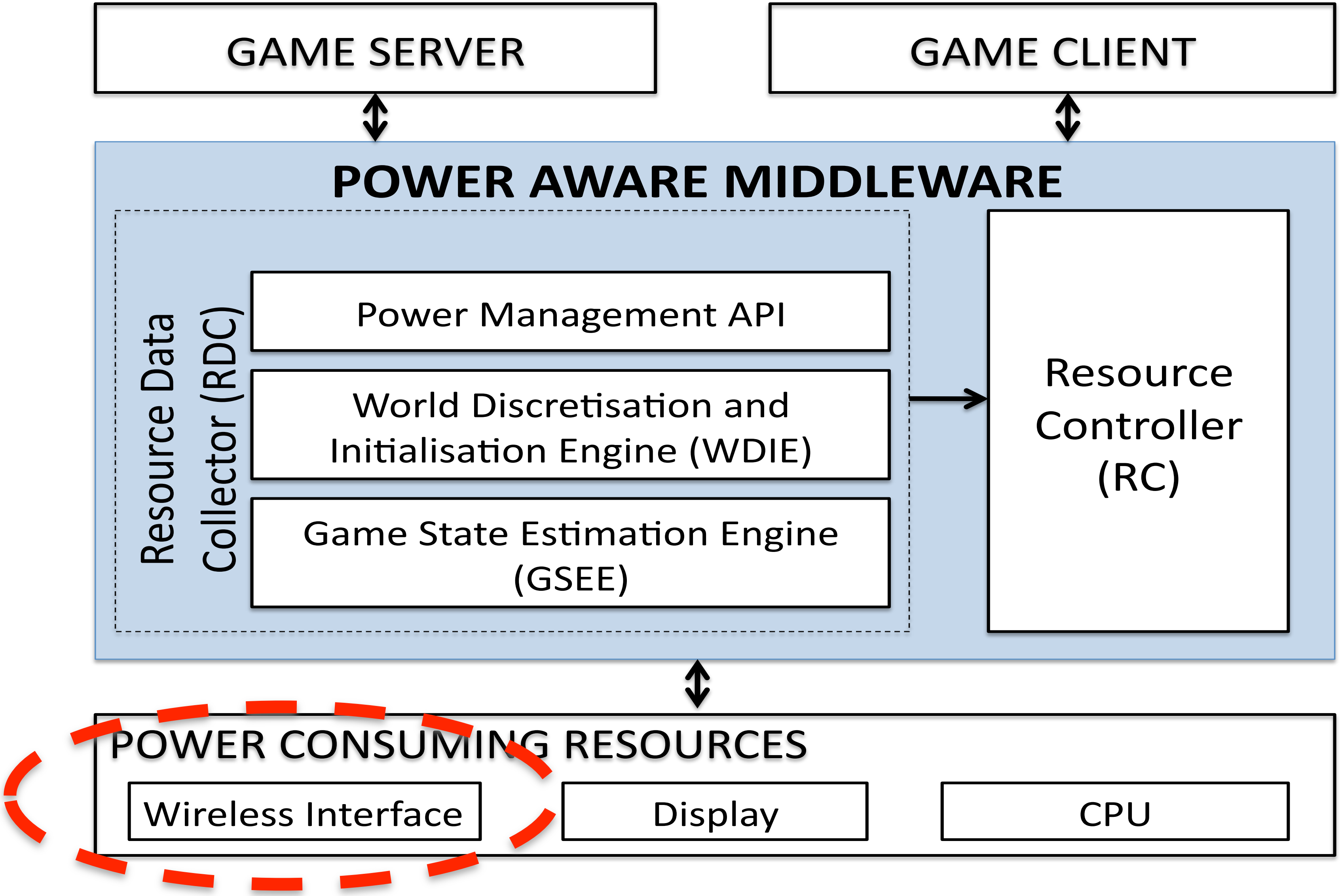
Measured on HTC Magic while streaming a Youtube Video

Middleware for Multiplayer Mobile Games

Design Objectives

- Should be power aware
  - Power is limited resource, its use should be reduced.
- Should be network aware
  - Wireless networks are unstable with high jitter, should tolerate it.
- Should Scale
  - Infrastructure should scale to Massive levels
- Should preserve quality of game play
- Should work for most of the game types, FPS, MMOG…

Power Aware Middleware - Architecture



Multiplayer Games

- Characteristics – FPS Games (Quake, Call of Duty 4, Counter-Strike…)
  - Highly interactive
  - 40-80 fps
  - 20-30 pps client to server (in regular interval, cannot burst!)
  - 40-60 pps server to client (in regular interval, cannot burst!)
  - Latency more than 150ms makes the game play annoying
  - Latency more than 300ms makes the game not playable. Most likely the server will disconnect

Multiplayer Games

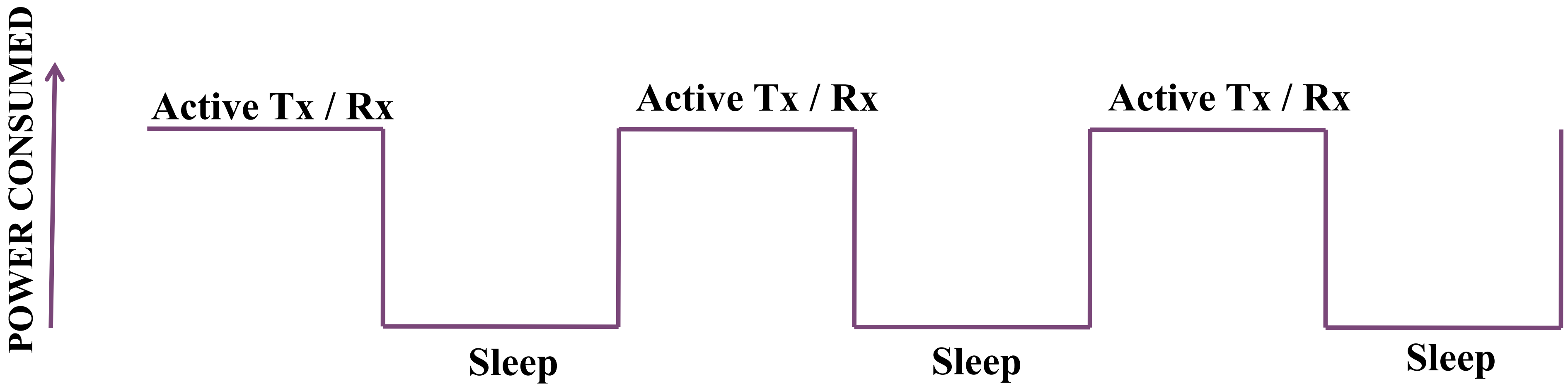
- Characteristics - RPG/Strategy game (WoW, Linage 2)
  - Combination of high & low interactive areas
  - 40-60 fps
  - ~10 pps client to server
  - 15-40 pps server to client
  - Can tolerate latency 400 – 600 ms, depends on the game state

Complexities

- Real-time AV streaming.
  - Can pre-fetch/buffer for smooth playback and energy management
  - Strong relation between successive frames, easy to interpolate
  - Not highly interactive
- Games
  - Strict real-time constraints
  - No much use of buffering and interpolating
  - Highly interactive

Power Savings On Network Interface - Basic Technique

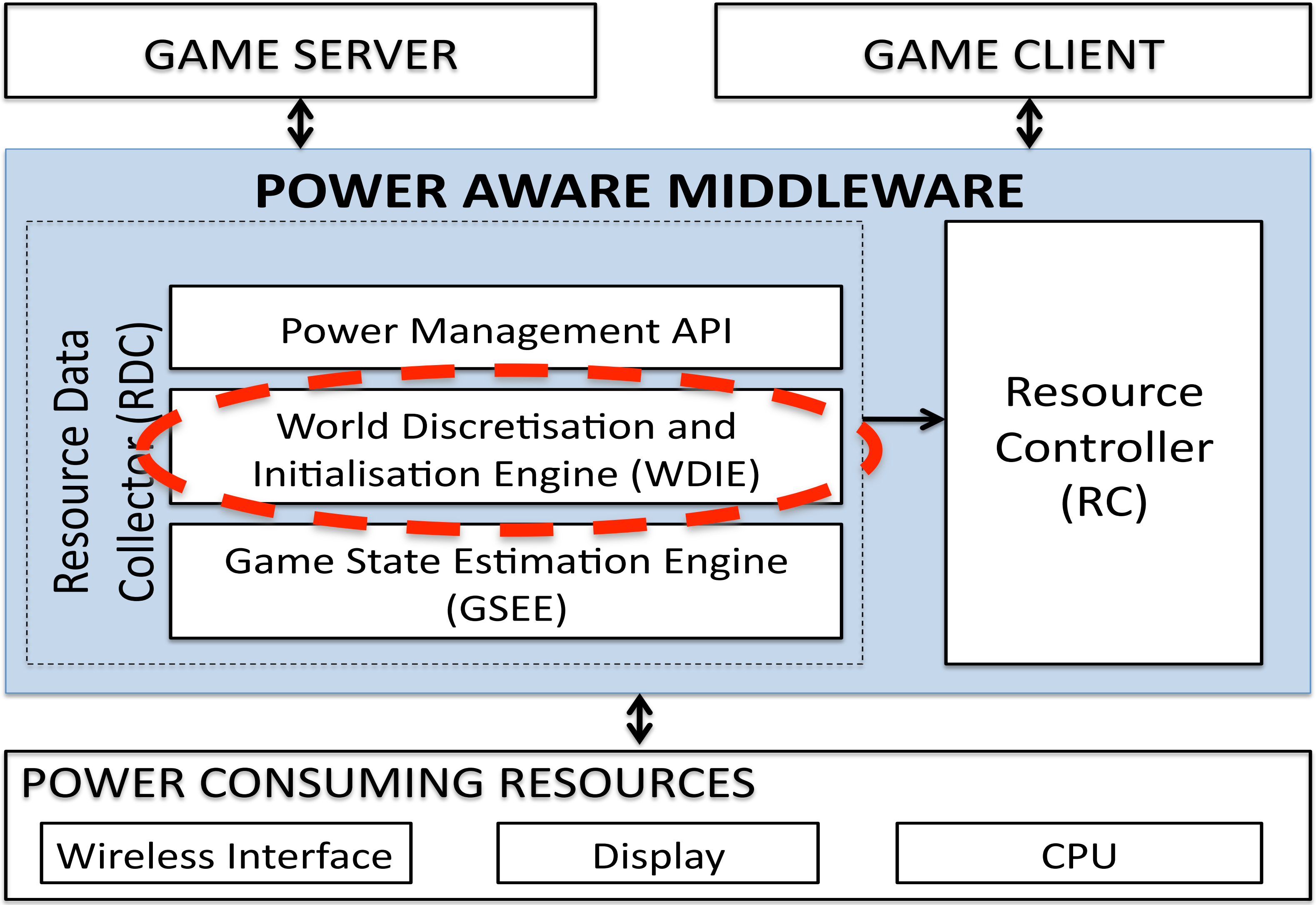
- Put Network Interface to sleep
  - Tx =Rx (~ 248 mA); Sleep ~ 4.66 mA
  - Current Schemes - Network Access Pattern and Application's Intent.
  - Games – Strict Real-time Requirements
    - When to put SLEEP mode ? and How long ?



One possible solution…

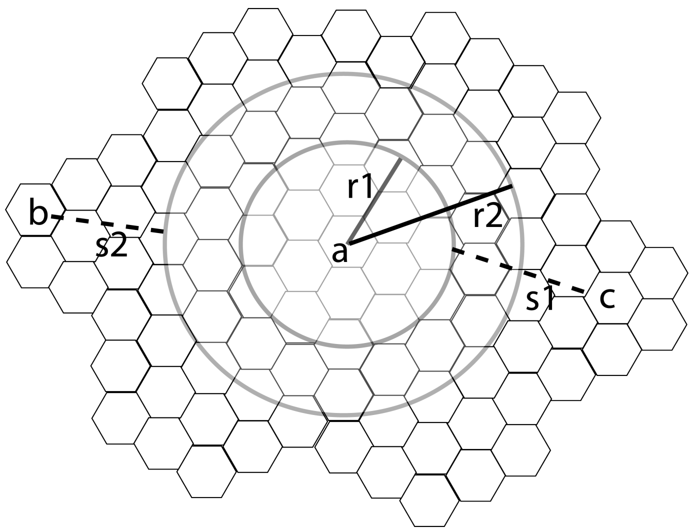
- SLEEP (for short periods) when game state/activity is not important
  - No loss of Important Packets
  - Missing game state can be extrapolated (DR, …)
- Implemented both on server and client side, to get maximum possible information
  - More information => high power saving with minimum loss to quality

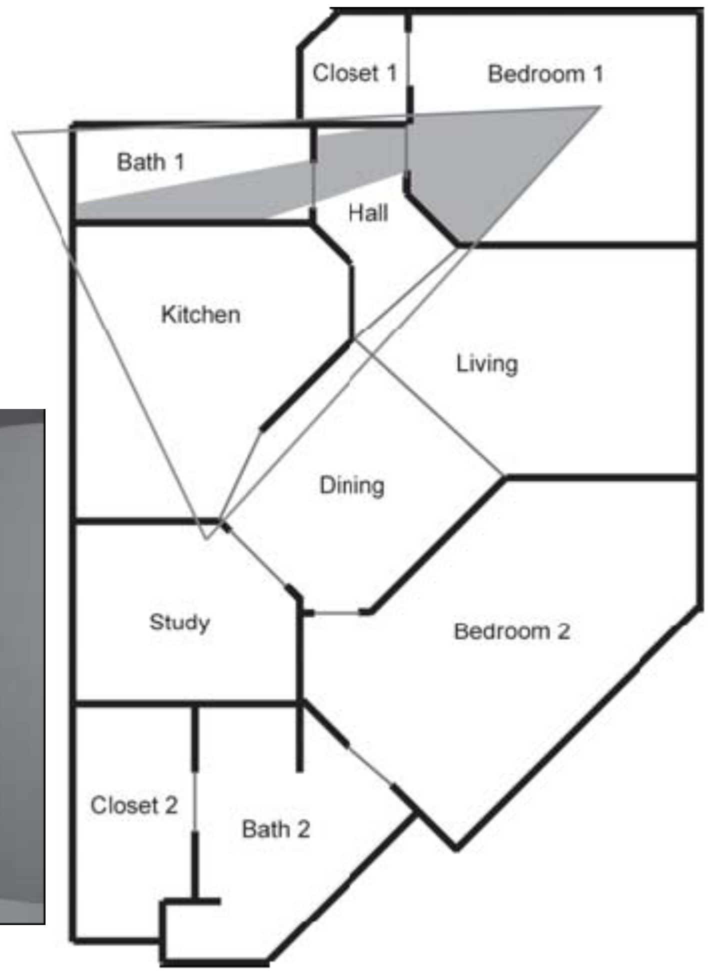
Power Aware Middleware - Architecture



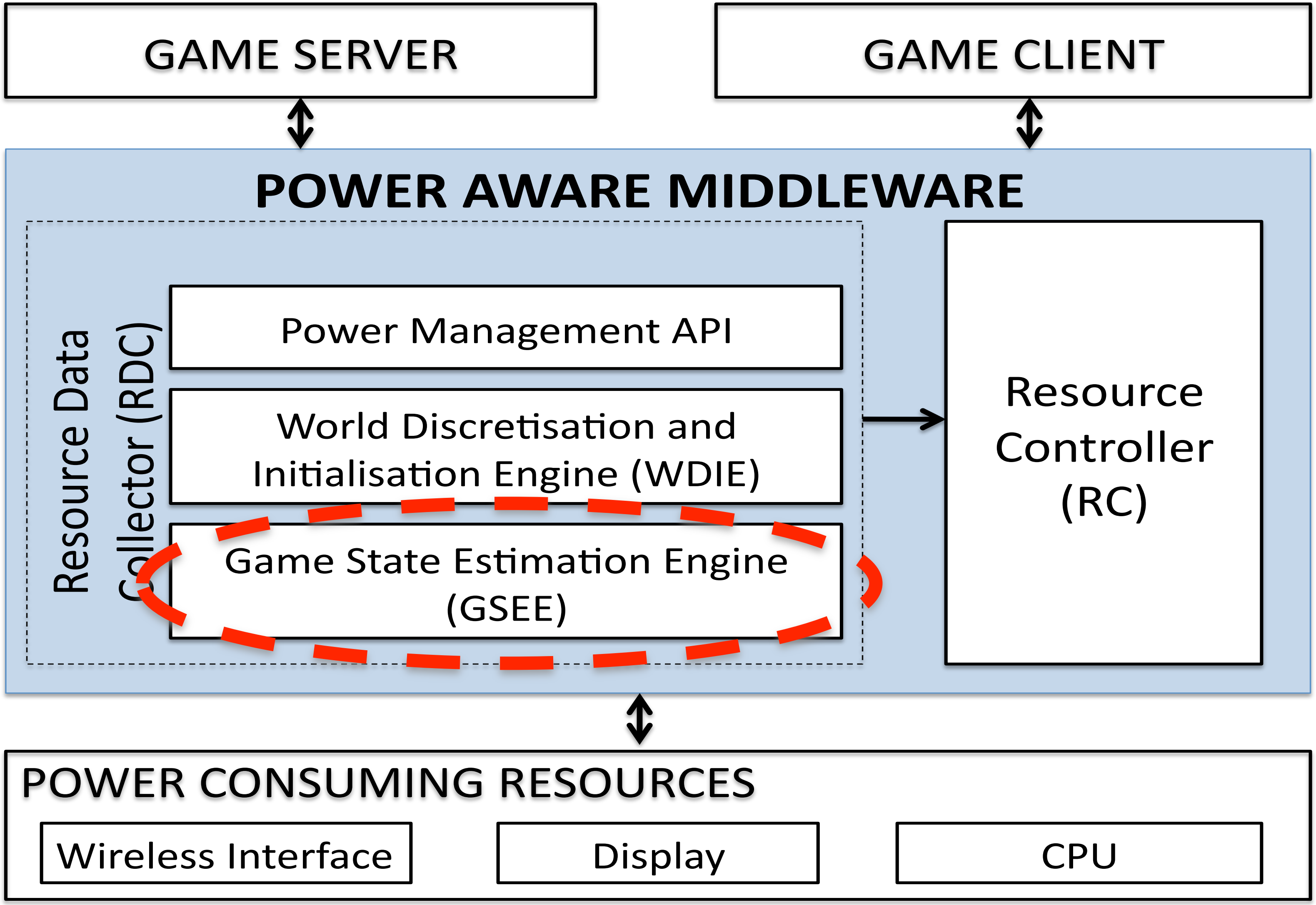
World Discretisation and Initialization

- Tile Based (mostly 2D)
- PVS based (Area, Cluster, Portal)   (3D)





Power Aware Middleware - Architecture



+

Game State Estimation Engine

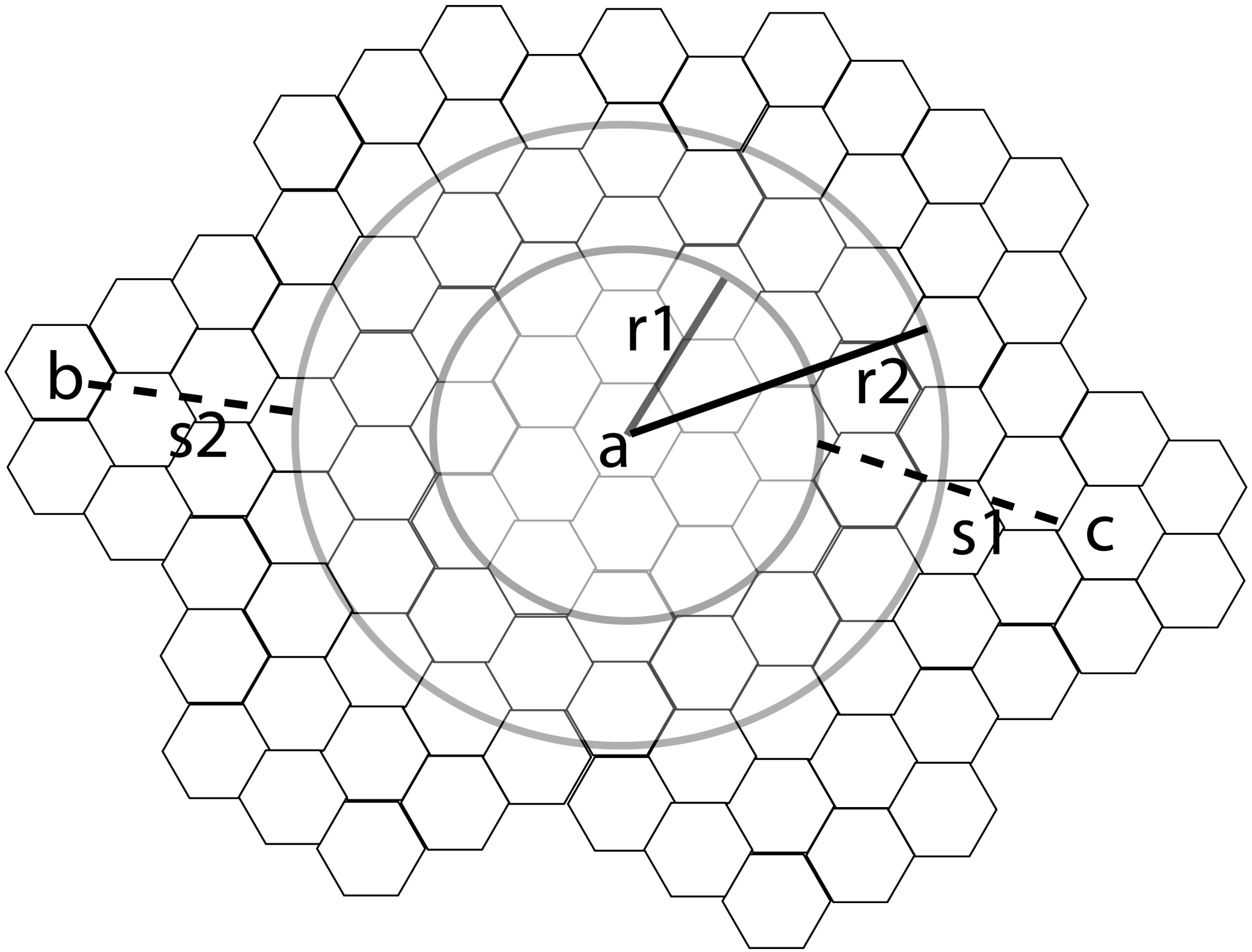
Various Algorithms (Requires Game Genre Based Selection)

- Macro Level (Server)
  - Distance Based
    - Single Ring
    - Dual Ring
    - Renderer's View Based - Path Distance (Area/Cluster/Portal) [Not Covered]
  - Visibility Based
    - Cell Based Visibility (2D)
- Micro Level (Server)
  - PAL
  - Game Action Prediction
  - Renderer's View Based – PVS (Area/Cluster/Portal) [Not Covered]

Algorithm – Single Ring

Relative Velocity Based

- MACRO level scanning (Server Side)



r – is dynamic based on the environment

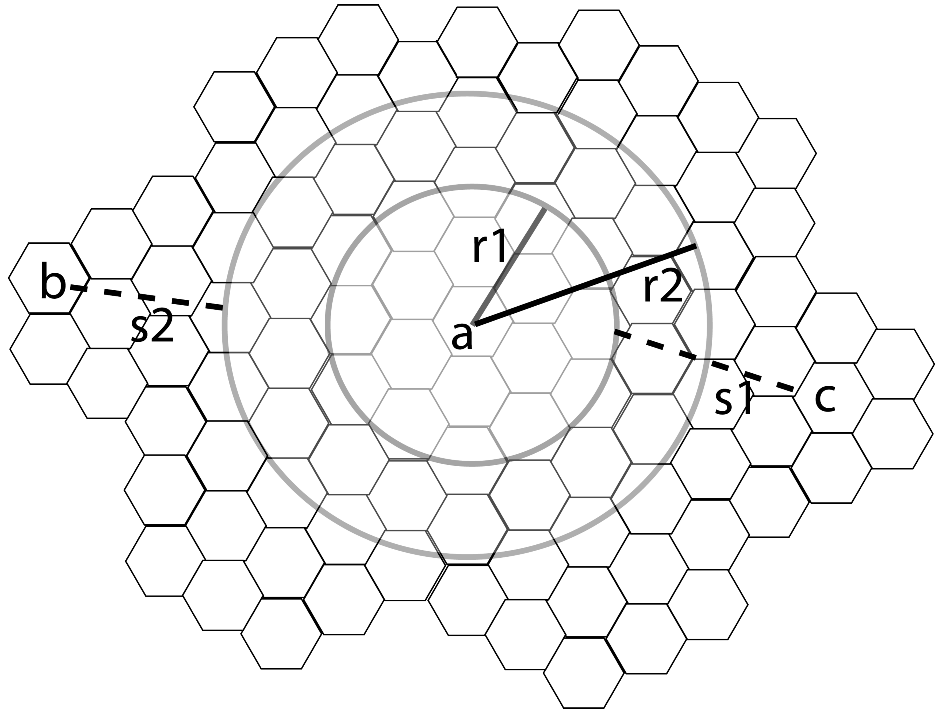
r = r1 or r2

r is the "vision range" of the player. How far the player can see clearly with higher LOD?

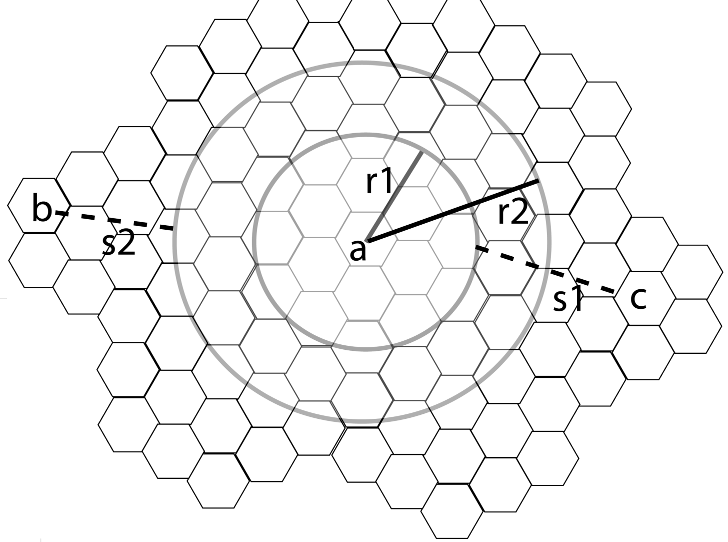
Algorithm – Single Ring

- MACRO level scanning
  - No entities in Vision Range => Non-critical state.
  - Desired state for WNIC SLEEP

- How long?
  - s_i => $Duration_i$
  - Smallest of ($Duration_i$)



+ Algorithm - Single Ring



1) **For each <interactive entity>**

- Calculate EuclideanDistance^2 to the entity
- Record history of Square of Proximities

2) **Select nearest 'n' entities**

3) **For each <nearest interactive entity>**

- Compute relative velocity (bi-directional) using history of Square of Proximities recorded in step 1
- PSD = (currentProximity – Vision Range) / relative Velocity
- If PSD = negative then exit

4) **ESD = Minimum of all PSDs**

- Constraint (minSLEEP < PSD < maxSLEEP)

PSD – Potential Sleep Duration

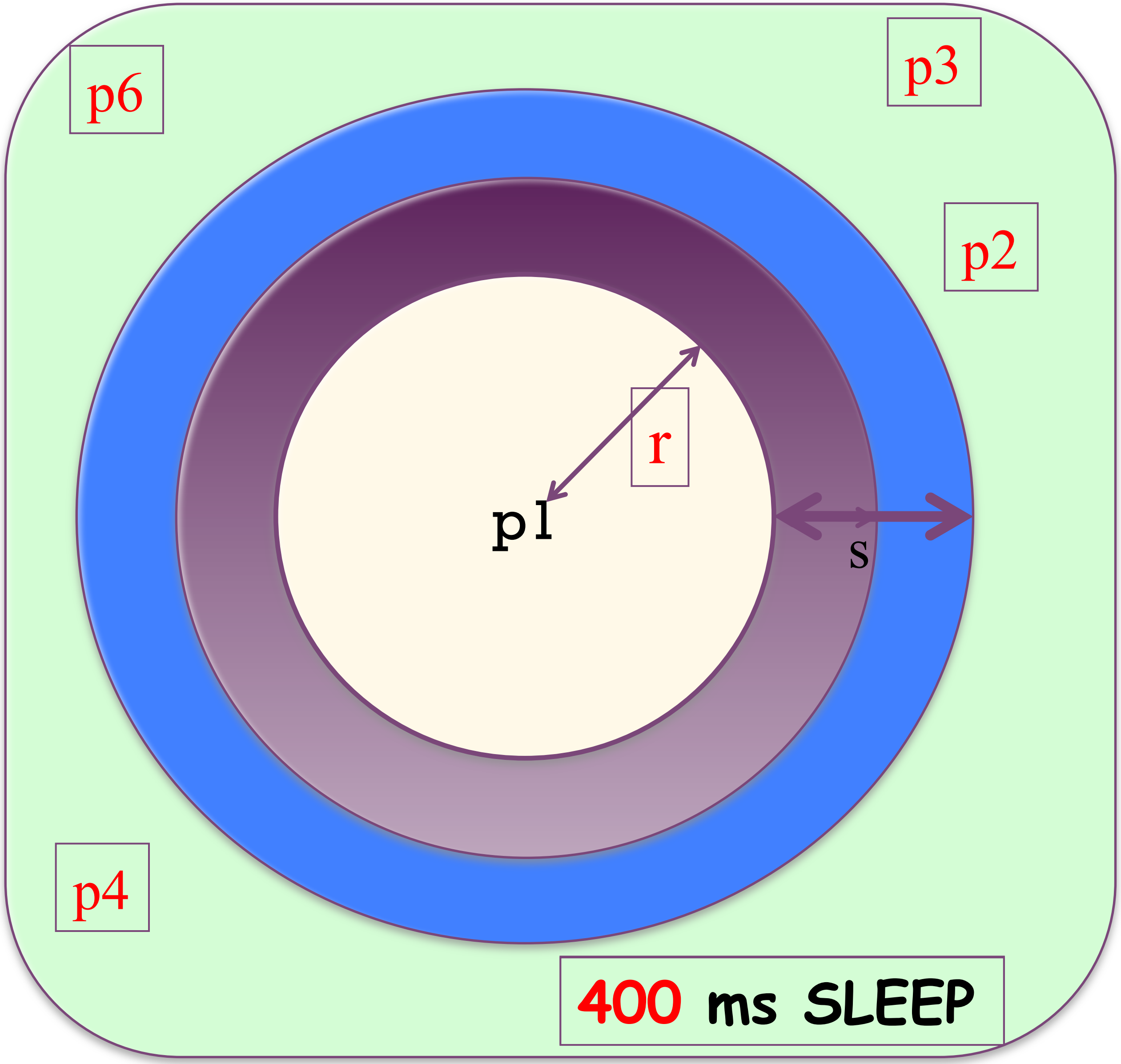
ESD – Effective Sleep Duration

+ Algorithm – Single Ring

- Can it scale?
  - Interaction Recency
    - RC maintains list of recently interacted entities for each client in Most Recent Interaction Table (MRIT) of size $m \times p$,
      - where m is number of clients and p is number of interactive entities a client is interested in.
  - Dual Ring
    - Games with high player density

+ Algorithm – Dual Ring

(Incremental Lookahead)



Vision Range (r) – is dynamic based on the environment

SafeArea (s) = r + (global average velocity of player x 200ms) .

's' grows in 200ms time steps… (upto 1sec for M^{3}ORPGs

Client's tile positions are registered with the Tiles!

Visibility Based Game State

- A state is important only if an opponent can see the player or vice versa.

  - Simplest Solution - If no opponent is visible, safe to sleep network card for some time.

Visibility Based Game State

- We need to make sure no opponent can reach the player while player NIC is sleeping

  - Advanced Solution – Predict ahead all possible movements and calculate max safe sleep time.

Visibility Based Game State

- Each player continues traveling in the direction he is traveling now, in the near future (100s of ms)

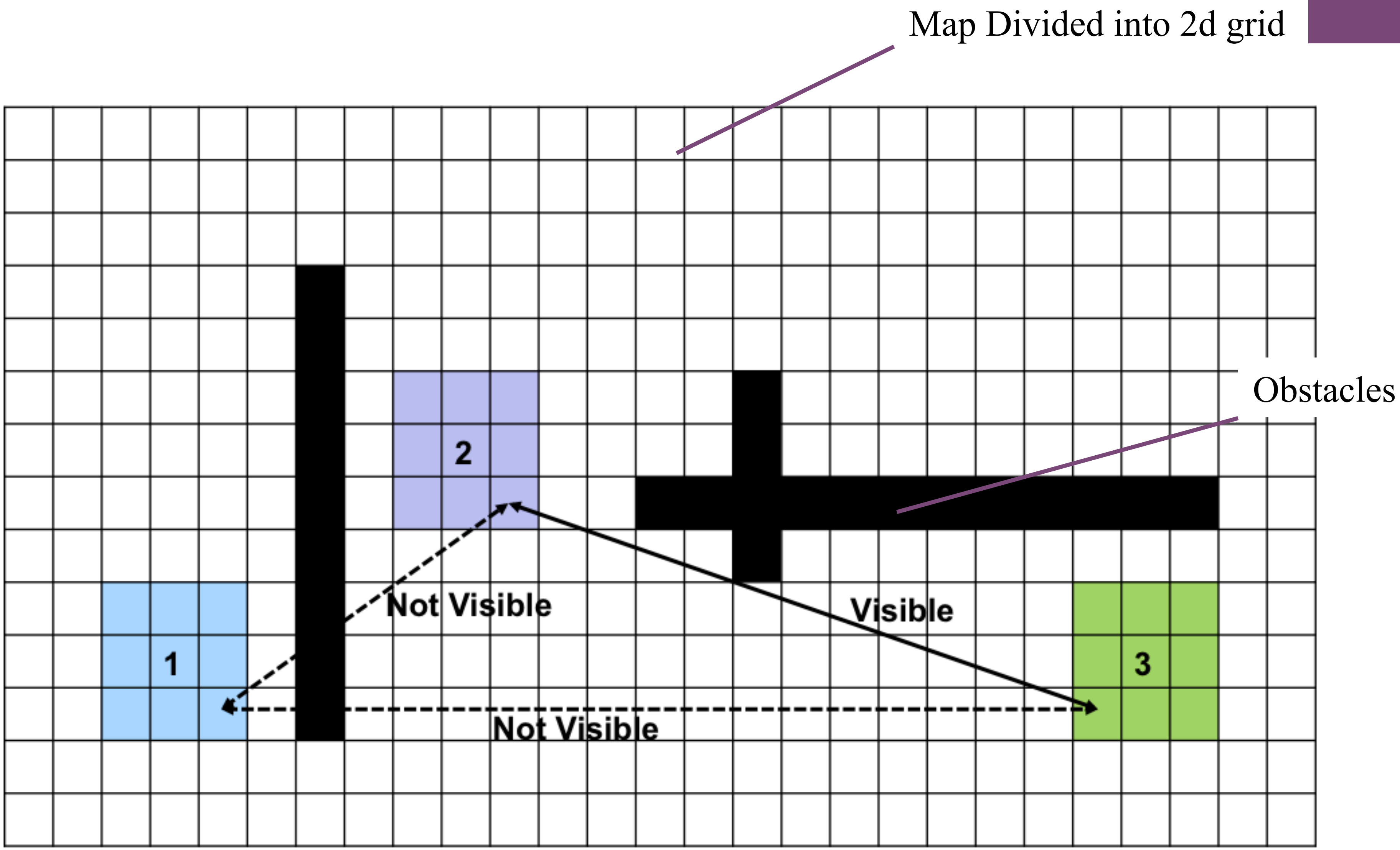
  - **Further Improvement** – Use direction based weights and error tolerance factor (α) for additional low risk sleep

Cell Based Visibility

- Game area is discretised into 2D grid
- Grid element size is related to MAX distance traveled in a set interval of time.
- Game provides function to check visibility between two points
- Used to pre-compute visibility between grid elements

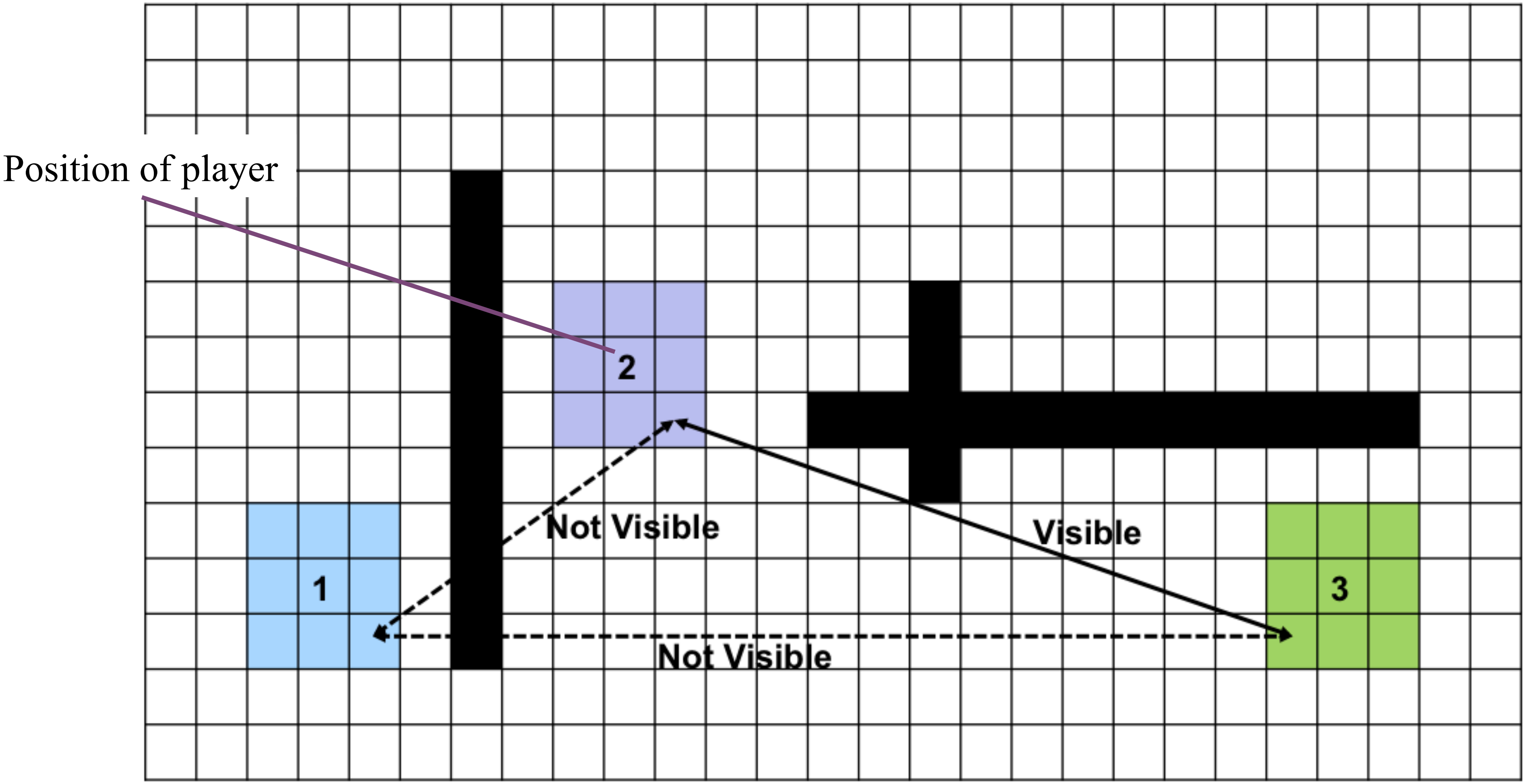
Cell Based Visibility

Look Ahead



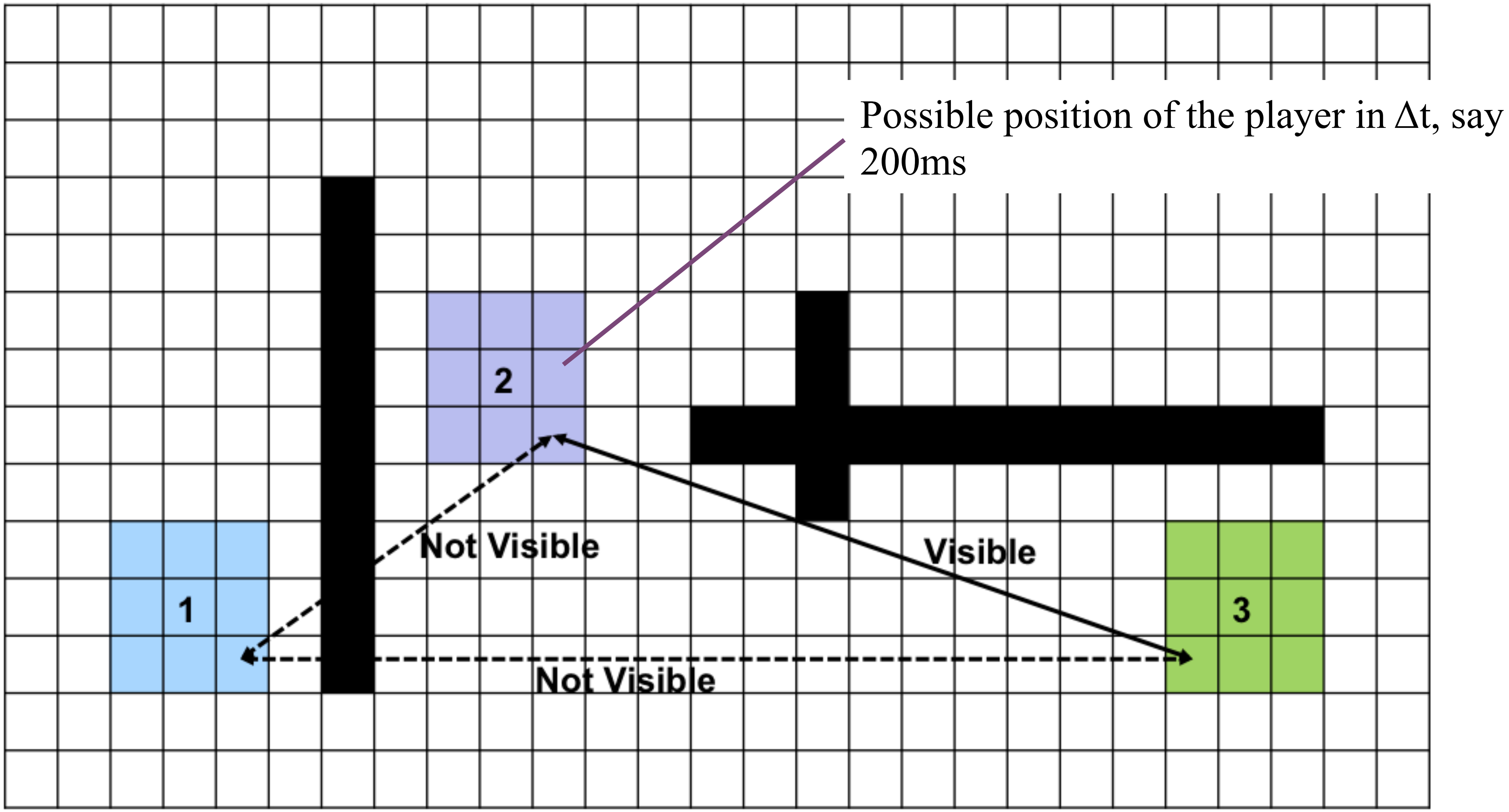
Cell Based Visibility

Look Ahead



Cell Based Visibility

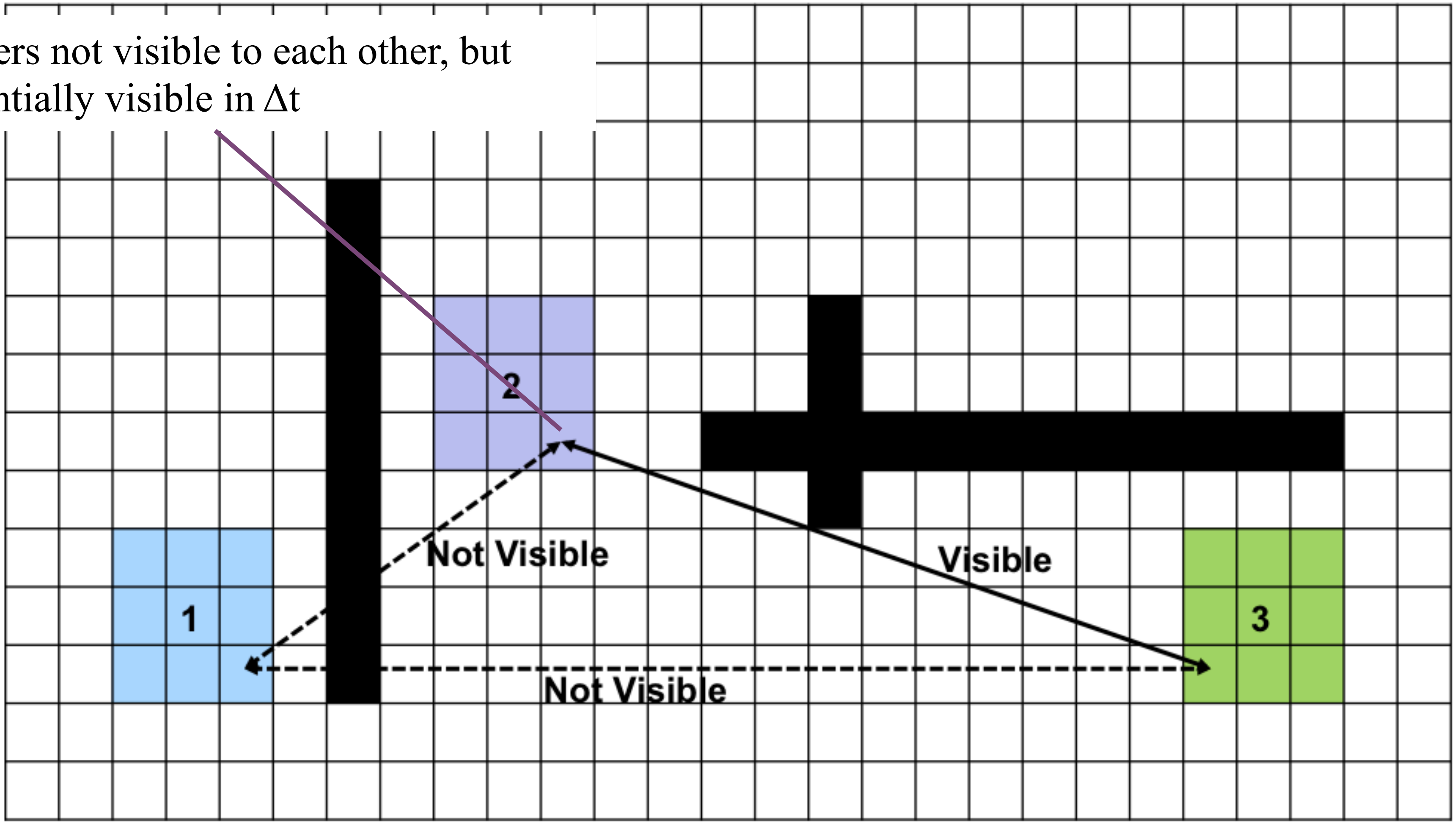
Look Ahead



Cell Based Visibility

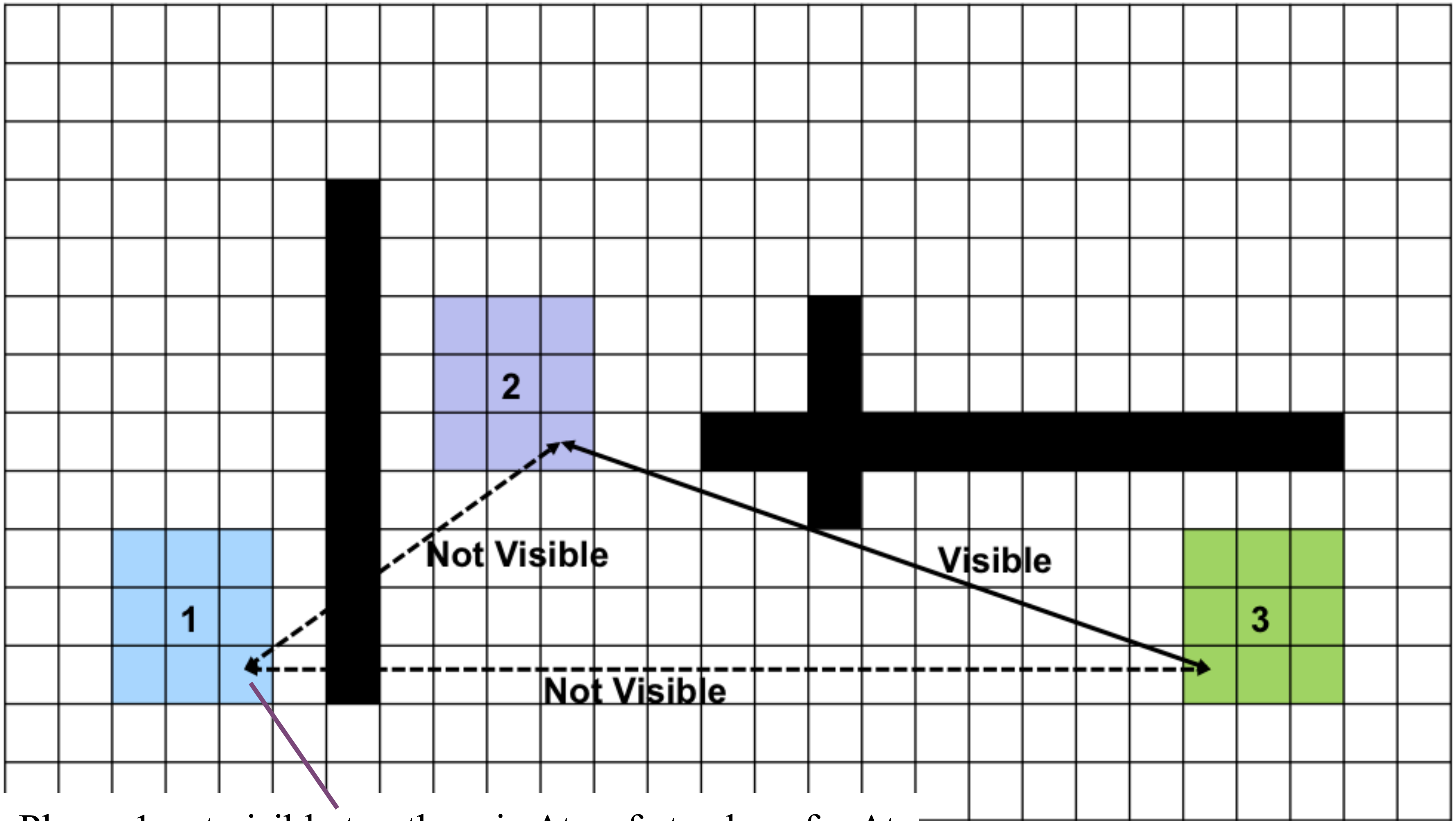
Look Ahead

Players not visible to each other, but potentially visible in Δt



Cell Based Visibility

Look Ahead



Player 1 not visible to others in Δt, safe to sleep for Δt

Cell Based Visibility

Normalized Sum of Costs

- Consider 2 players at grids x1 and x2
- In t time steps, let the grids reachable for players 1 and 2 be L_t^1 and L_t^2 respectively.
- For each point $l_1 \in L_t^1$ check its visibility to each point in L_t^2

$v(l_1, l_2)$ = 1 or 0

Based on Pre-Computed Visibility Matrix

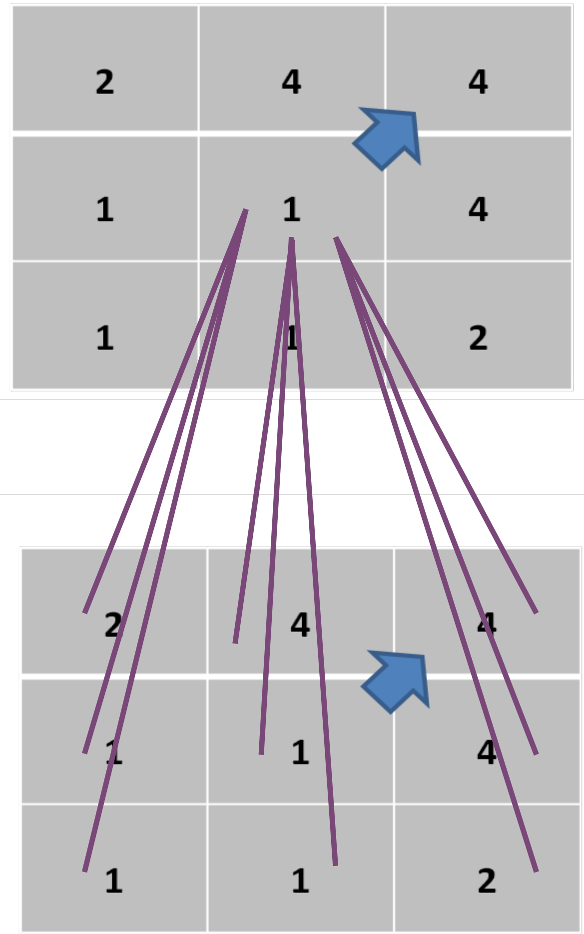
COST

$$S_t(1,2) = \frac{\sum_{l_1 \in L_t^1, l_2 \in L_t^2} w_{l_1} * w_{l_2} * v(l_1, l_2)}{\sum_{l_1 \in L_t^1, l_2 \in L_t^2} w_{l_1} * w_{l_2}}$$

Cell Based Visibility

Direction based Weights

- Different future grid positions assigned weights based on direction
- Calculate normalized sum of product of weights for each position that is visible, for 2 players
- If sum < error tolerance (α), safe to sleep



Direction based Weights

Different Sleep Times

- From a grid calculated to Δt intervals, algorithm can easily be extended to 2*Δt, 3*Δt.

- Previous steps first run on 3*Δt, down to Δt to calculate safe sleep time.

Potential player position in

3*Δt

2*Δt

Δt

(-3,3)	(-2,3)	(-1,3)	(0,3)	(1,3)	(2,3)	(3,3)
(-3,2)	(-2,2)	(-1,2)	(0,2)	(1,2)	(2,2)	(3,2)
(-3,1)	(-2,1)	(-1,1)	(0,1)	(1,1)	(2,1)	(3,1)
(-3,0)	(-2,0)	(-1,0)	(0,0)	(1,0)	(2,0)	(3,0)
(-3,-1)	(-2,-1)	(-1,-1)	(0,-1)	(1,-1)	(2,-1)	(3,-1)
(-3,-2)	(-2,-2)	(-1,-2)	(0,-2)	(1,-2)	(2,-2)	(3,-2)
(-3,-3)	(-2,-3)	(-1,-3)	(0,-3)	(1,-3)	(2,-3)	(3,-3)

Cell Based Visibility

Determining Sleep Duration

For each player i

For each time slot $t \in T$

For each player $j \in N \setminus i$

compute $S_t(i,j)$

t is feasible if $S_t(i,j) \leq \alpha, \forall j$

Select the largest feasible t as user i's sleeping interval,

otherwise user i does not sleep.

Alpha varies form 0 – 1.
0 – conservative
1 – aggressive (always sleep)

t – is potential sleep duration (PSD)

Intro to Renderer's View/PVS Based

Basics

- 3D spatial subdivision is used for collision detection, rendering…
  - The world is split into small convex hulls (areas). From the convex hull, there is in fact a limited number of other convex hulls can be seen.
- Potentially Visible Sets (PVS) is a set of potentially visible areas form current area. It is map dependent and pre-computed.
  - For occlusion culling, used by renderers, pre-computed, then indexed at run-time in order to quickly obtain an estimate of the visible geometry.
  - It is symmetric
- We leverage on this in-game spatial subdivision & PVS (lower overhead costs)

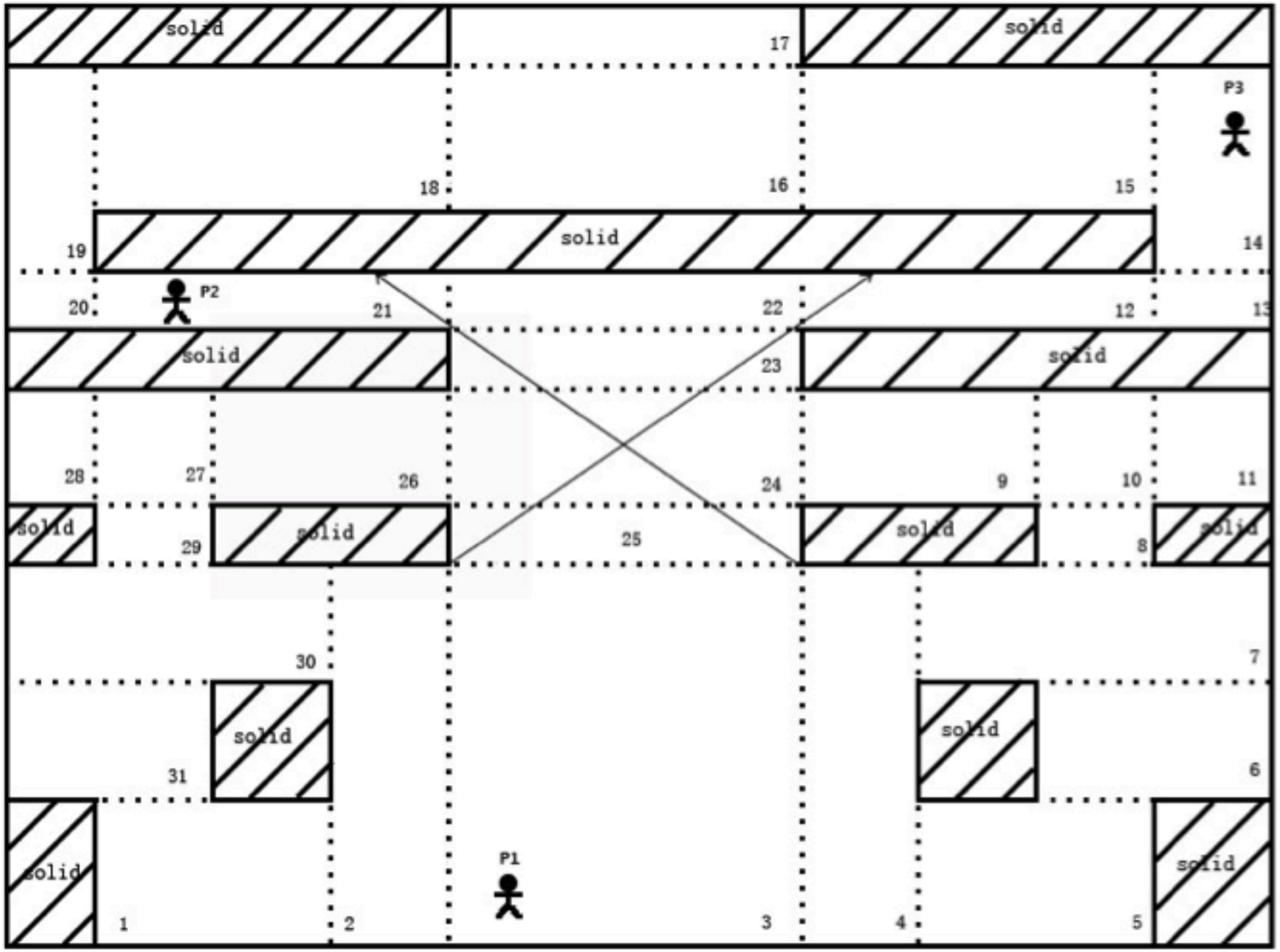
Intro to Renderer's View/PVS Based

Background: Cluster, Area and Portal

- Adjacent areas shared same surfaces are grouped together to form a cluster.
  - Eg. Room is a Cluster with areas for walking, area under table…
- The door connecting two clusters is a cluster portal.
  - Some modern games allow the artist to specify cluster portals.

Intro to Renderer’s View/PVS Based

PVS in one Cluster



If it is translucent object, vision can go beyond

Can be fully computed

<or>

Artist can specify Portals to simplify

Intro to Renderer's View/PVS Based

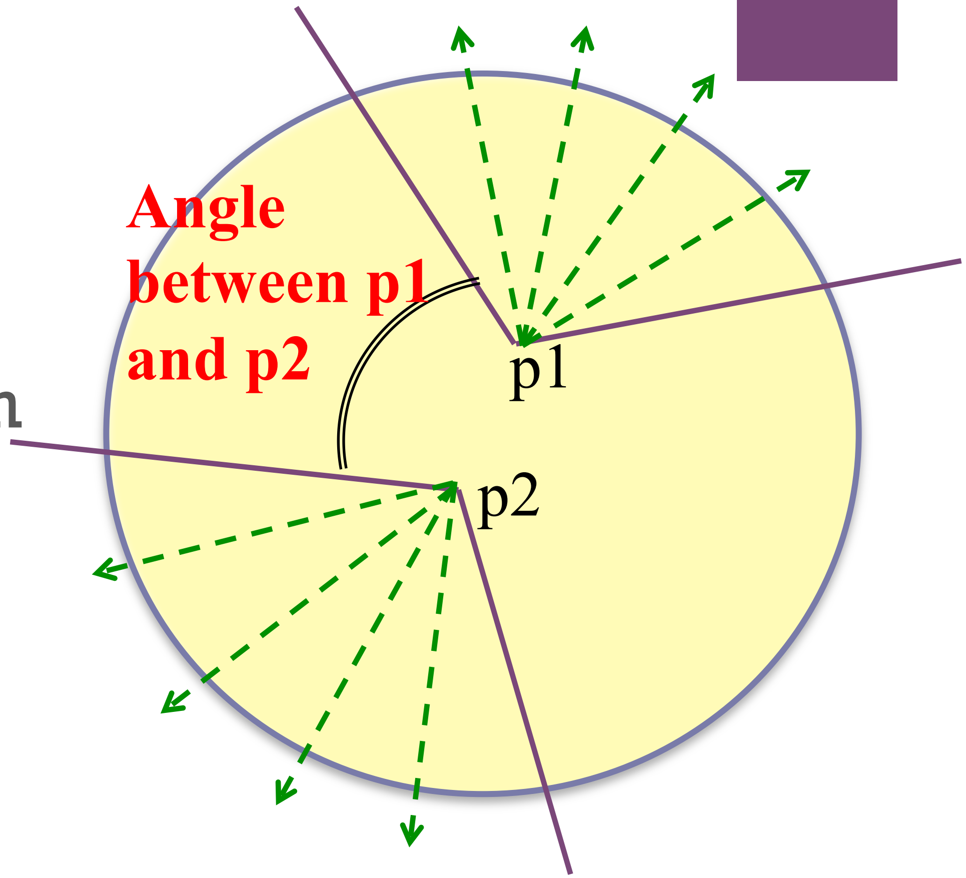
Renderer's View of the World (PVS)

- Safe Period: When No other players in the current player's PVS set.

Micro Level Algorithms

Server Side

- When there are interactive entities inside 'vision range'
- Multiple Schemes – based on availability of data
- Server side
  - field of view of a character is around – 2 * PI/3
  - Max turning speed – 2 rad/second or 0.5 rad/200ms



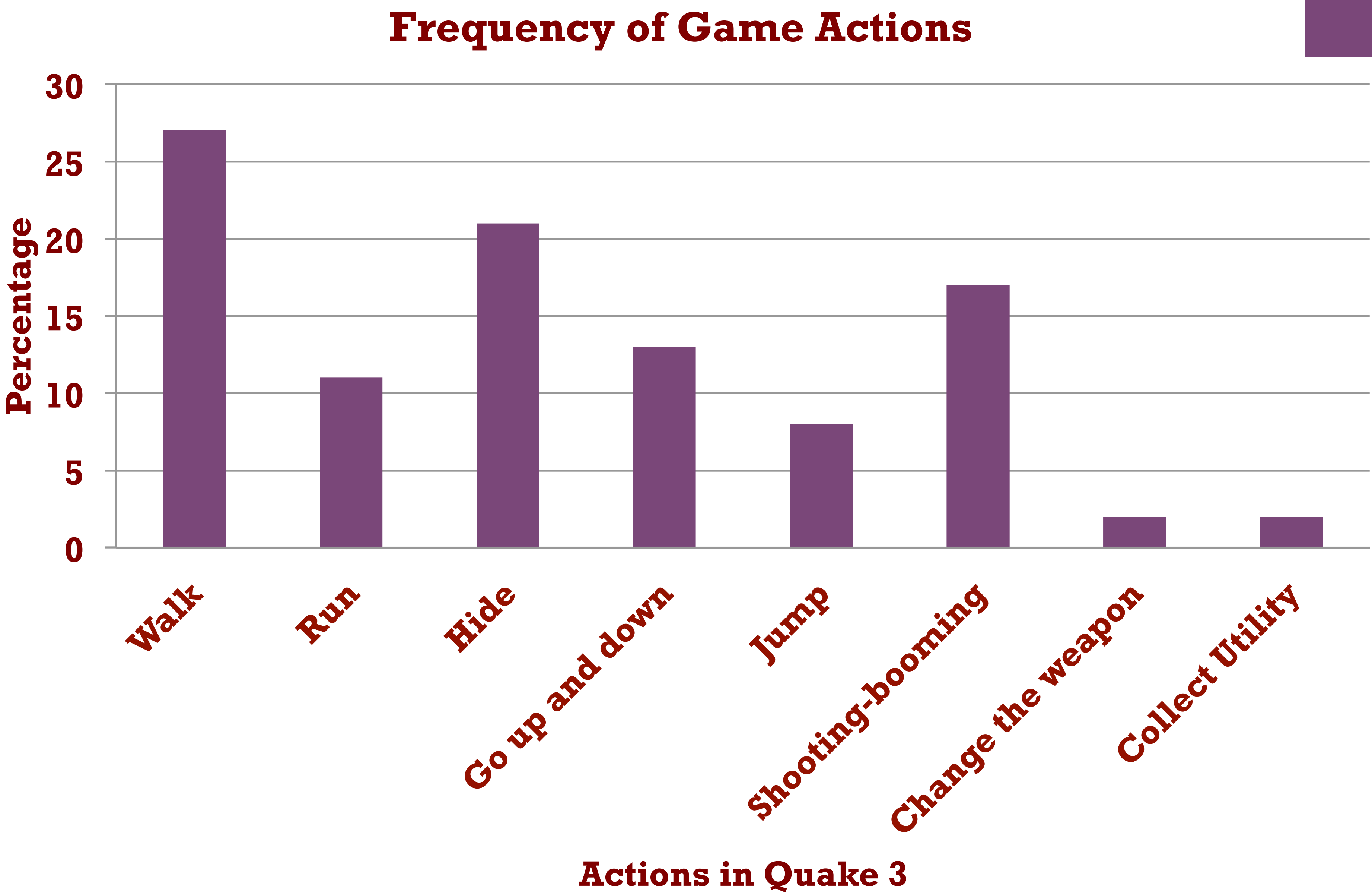
Micro Level Algorithms

Client Side (Only)

- PAL (Player Activity Level) prediction
  - Def: No of key_press and mouse events per second
  - Correlation between PAL and Game State
  - State is critical if PAL > threshold
  - PAL_threshold is set based on the player's expertise level
  - Can be measured using the data from game or externally as an independent tool
- Prediction is based on weighted historical data, with high weight for most recent data

Micro Level Algorithms

Client Side (Client Only)
(Game Action Prediction)



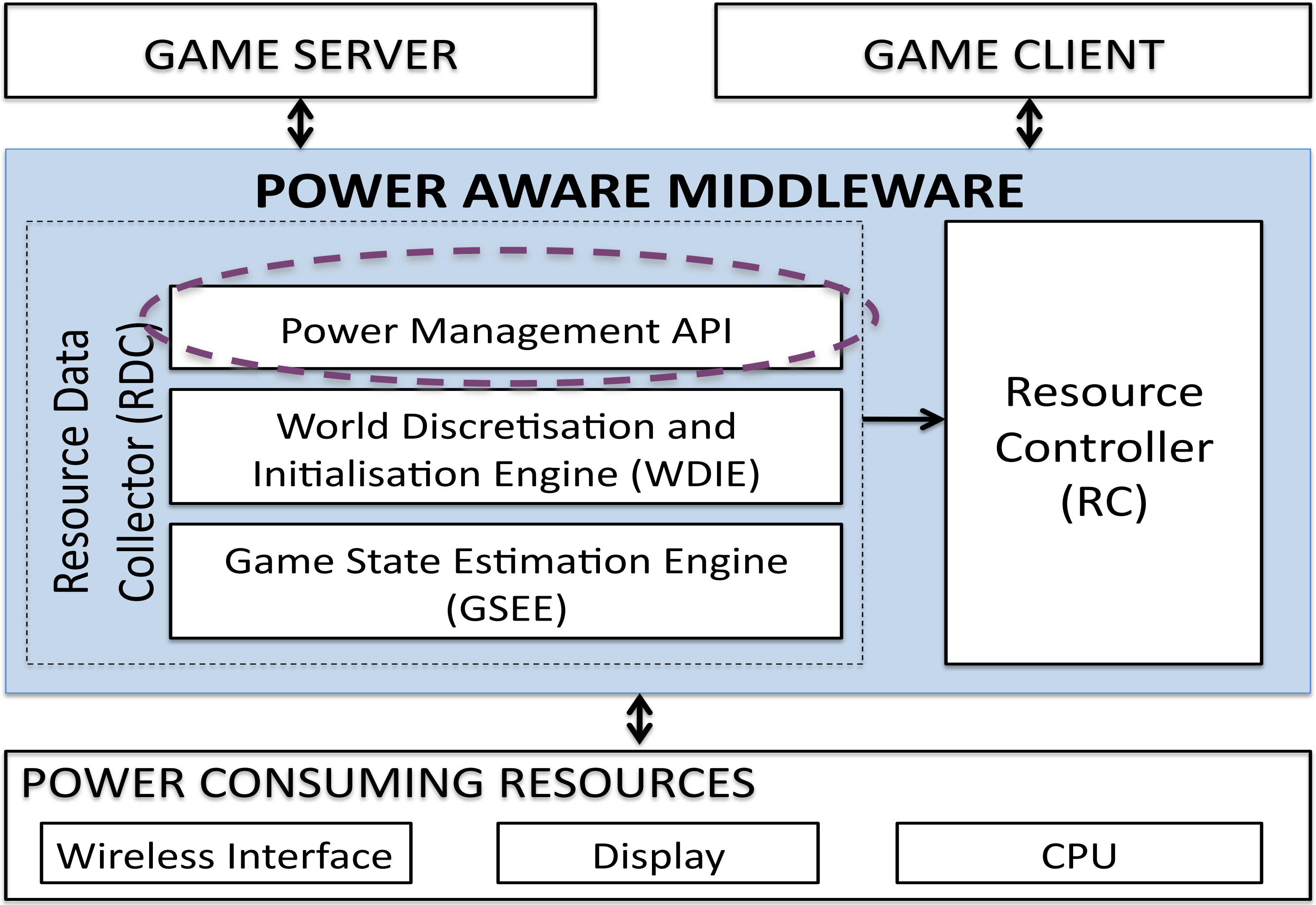
Micro Level Algorithms

Client Side (Client Only)

- GAP (Game Action Prediction) Engine
  - There is a correlation between the game action and game state.
  - ARIVU currently captures: Idle, Attacking, Moving, Accessing Menu, Dead, Chat, Trading, Item Interaction, Interacting with other avatars, Interaction with NPC

- Prediction is based on past history of actions
  - $GameAction(i+1)=w_j$ * GameAction(i-j); f or j =0 to n-1
  - $w_0 > w_1 > w_2 > w_3 > w_4 > \ldots\ldots > w_{n-1}$
  - Initial weights are 1/2, 1/4, 1/8, 1/16 and 1/16 for w_0 to w_4

Power Aware Middleware - Architecture



RDC collects the following through API …

- Server Side:
  - Game map info [size and shape]
  - Positions of entities
  - Entity interactions
  - Game environment
  - Game player expertise level
  - Game genre (To select appropriate MACRO / MICRO algorithm)
- Client Side:
  - Key press and mouse events (interactions)
  - Game actions of players

+

Implementation & Results

Armageddon RPG

Quake III

FPS

Ryzom

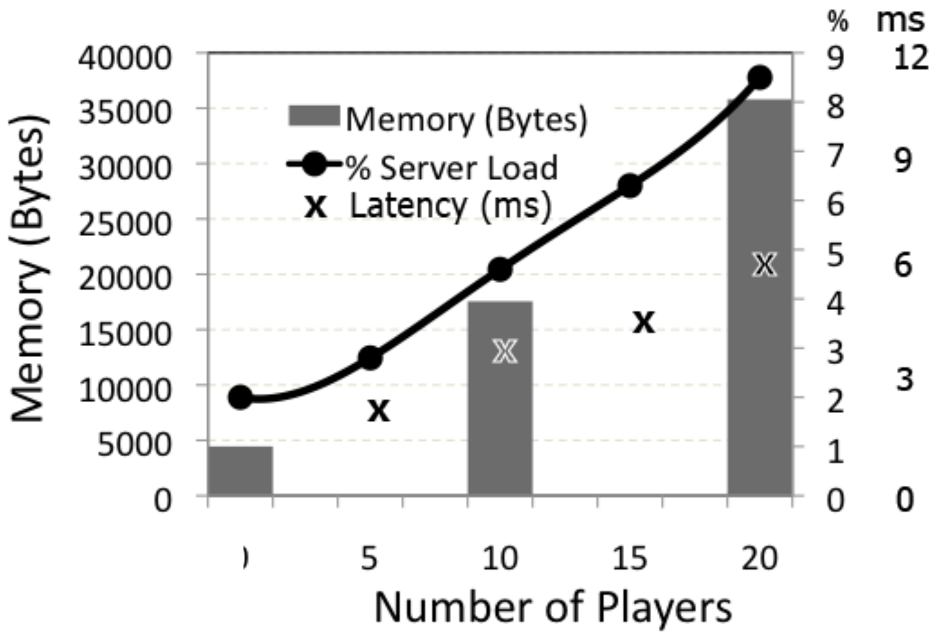
MMORPG

Armageddon (RPG)

+ ARIVU on Armageddon

Built our own Android game, an isometric Mobile Multiplayer RPG called Armageddon. (FPS mode can be simulated)

Algo: Single Ring (Macro), View angle (Micro).



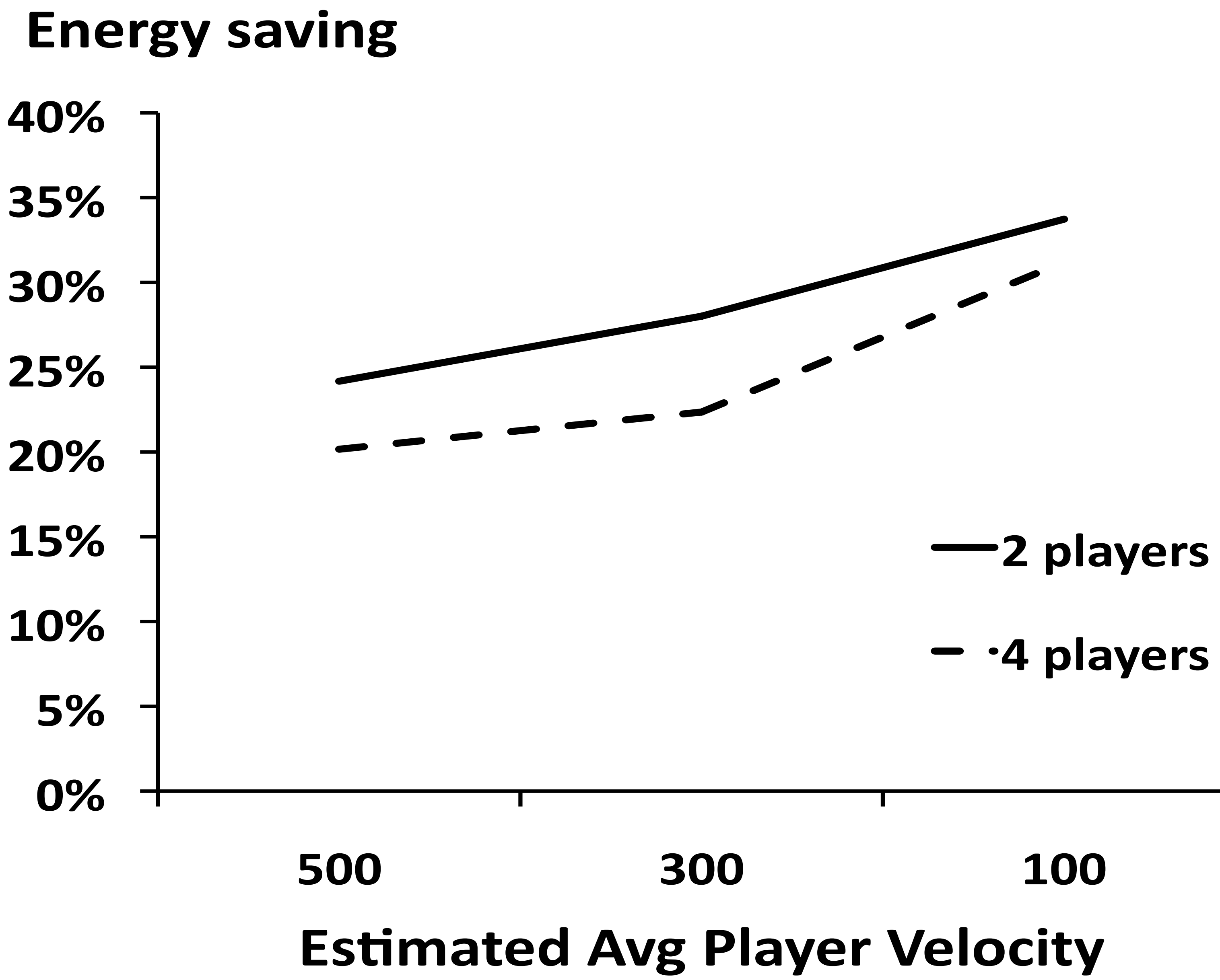
CLIENT SIDE IMPACT

Metric	MW Enabled	MW Disabled	Overhead
Current Drawn by CPU(mA)	213.39	207.29	6.1
Memory Used(KB)	2294.6	2293.1	1.5

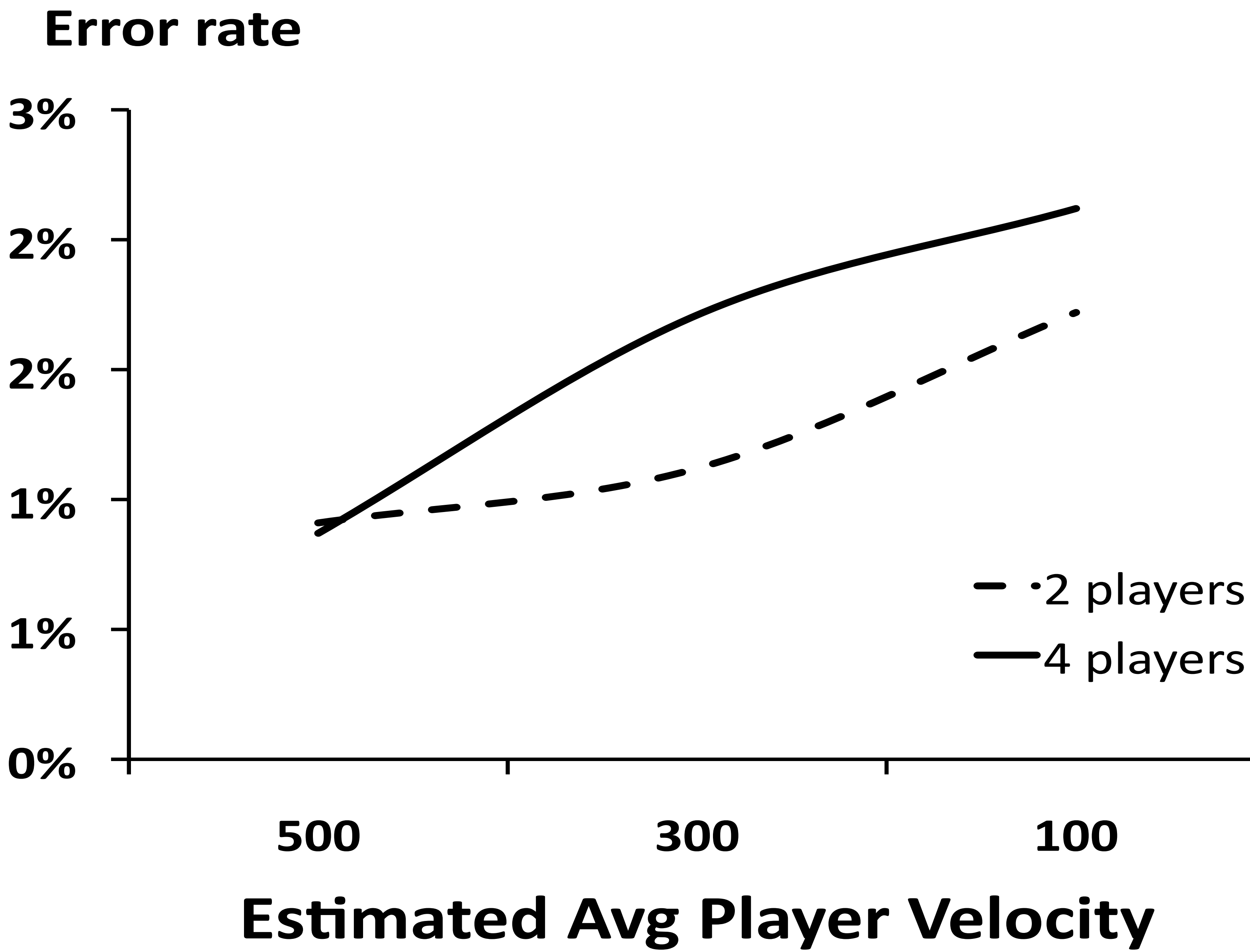
Evaluation

- The effective "vision range" for friendly environment is 125 pixels and hostile area is 250 pixels.
  - All the variants are tested with 6 human players and 3-12 bots. Interaction recency is used to boost the scalability (instead of Dual Ring)
- A packet is important for a client if, when the packet is transmitted, there is at least one interactive object within its vision range with which there is at least one interaction.

+ Results (RPG)

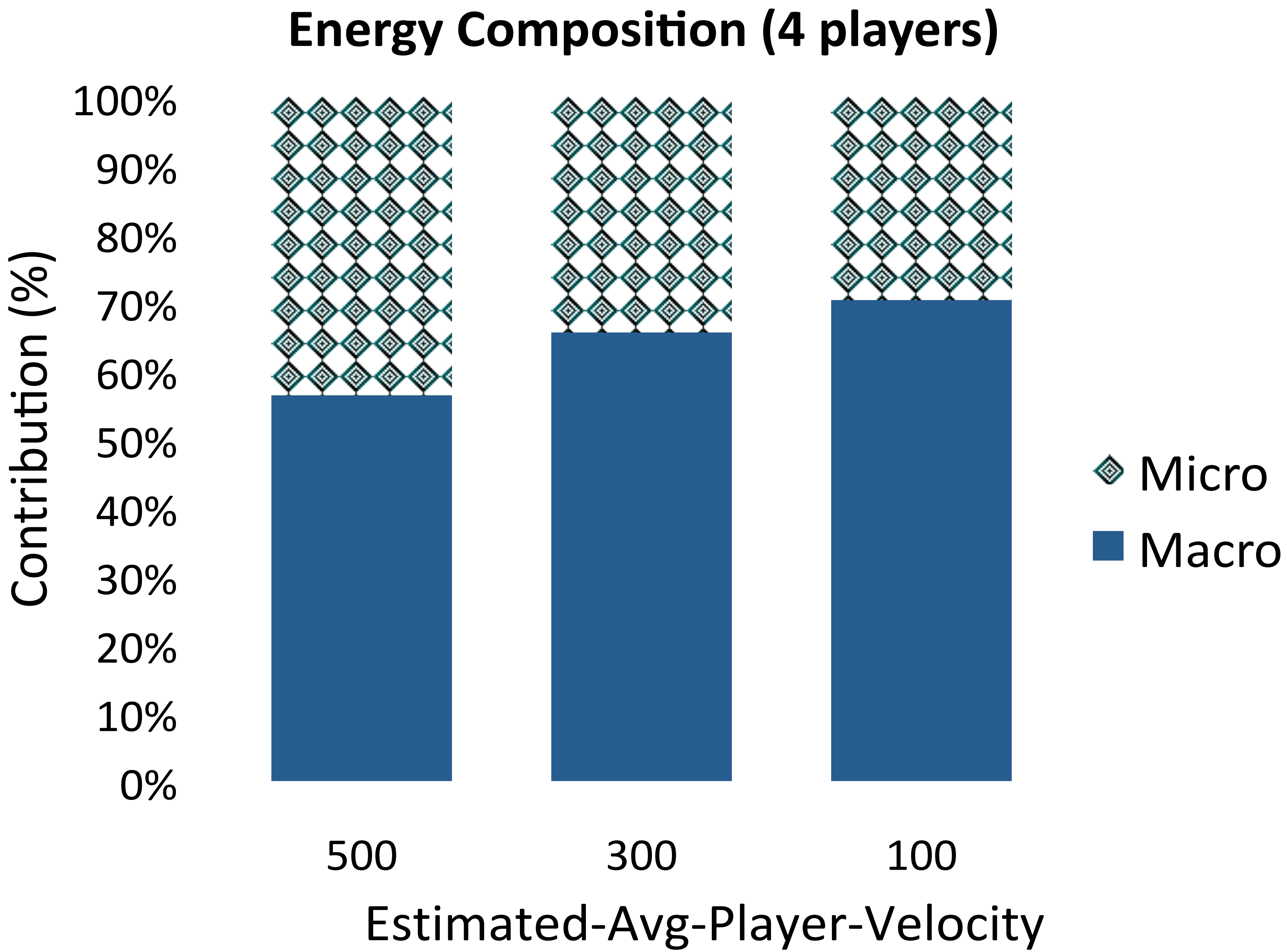

Energy saving
40%
35%
30%
25%
20%
15%
10%
5%
0%
2 players
4 players
500
300
100
Estimated Avg Player Velocity

Results (RPG)



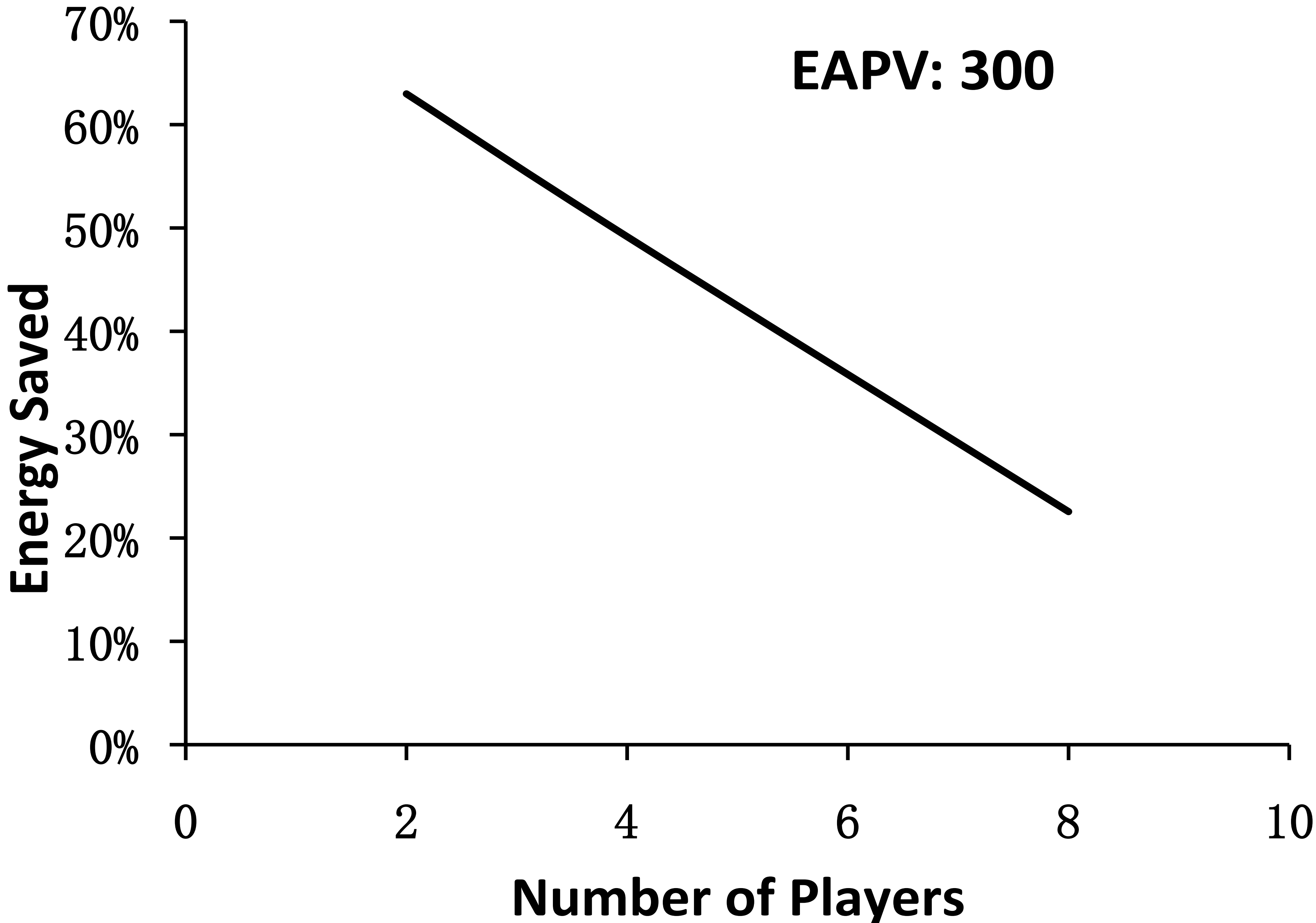
Error rate
3%
2%
2%
1%
1%
0%
2 players
4 players
500
300
100
Estimated Avg Player Velocity

+ Results(RPG)

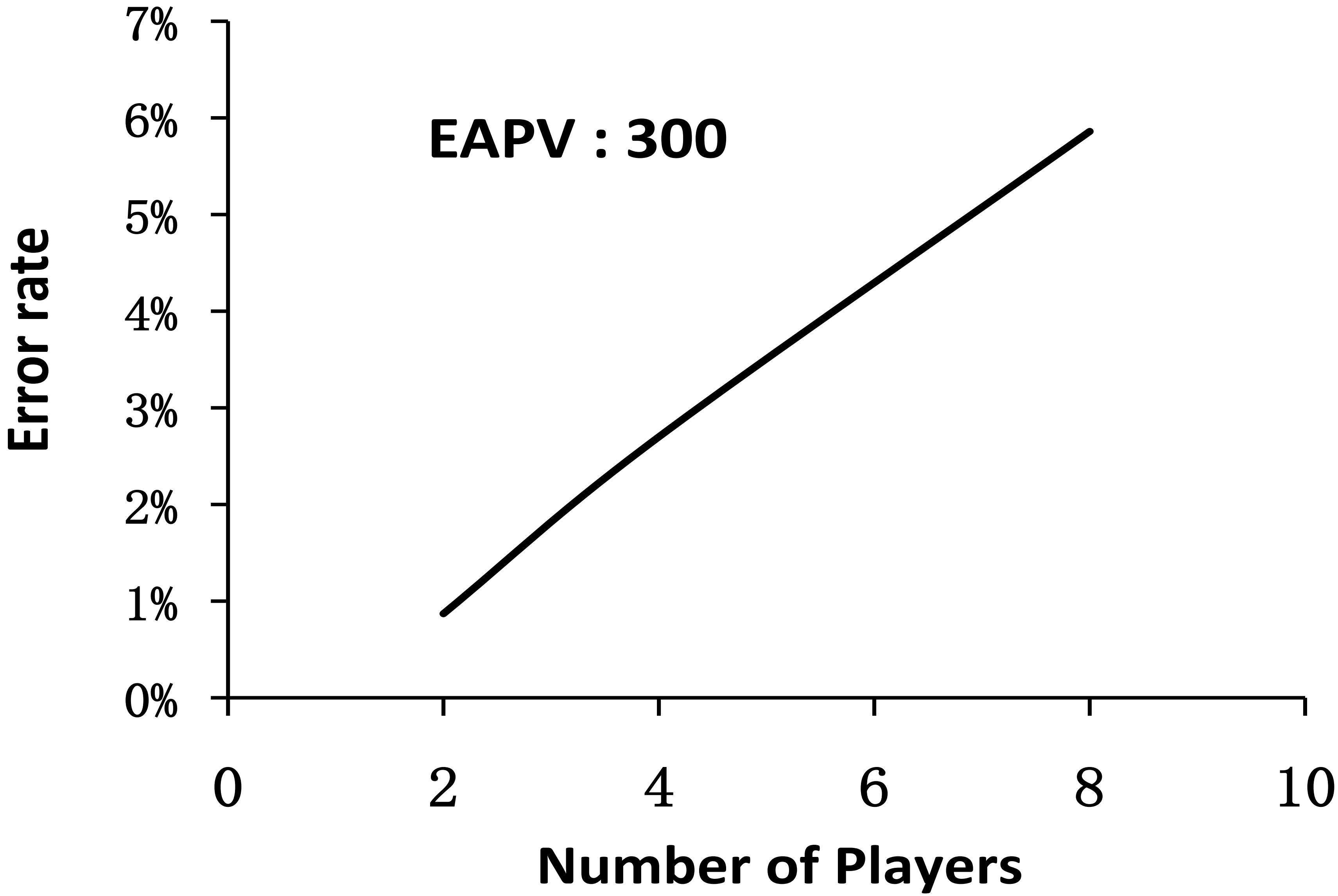


Energy Composition (4 players)
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Contribution (%)
500
300
100
Estimated-Avg-Player-Velocity
Micro
Macro

+ Results (Simulated FPS with More Players)



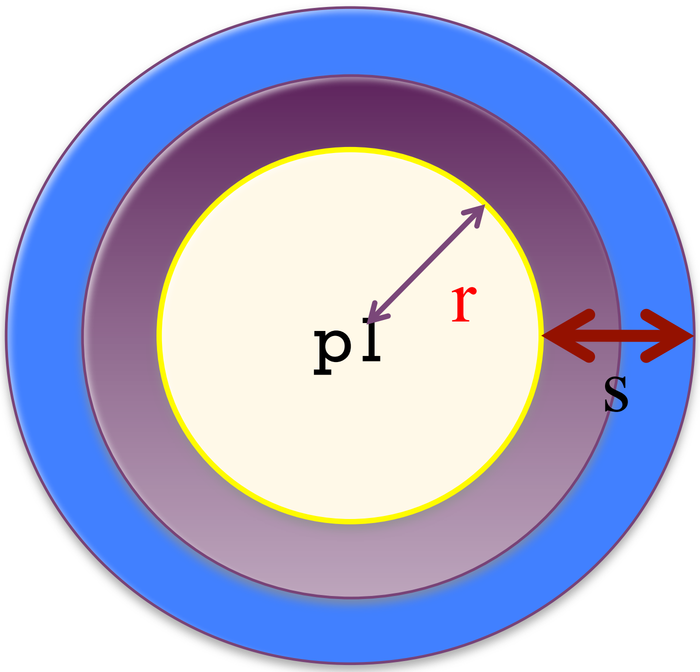
Results (Simulated FPS) with more players

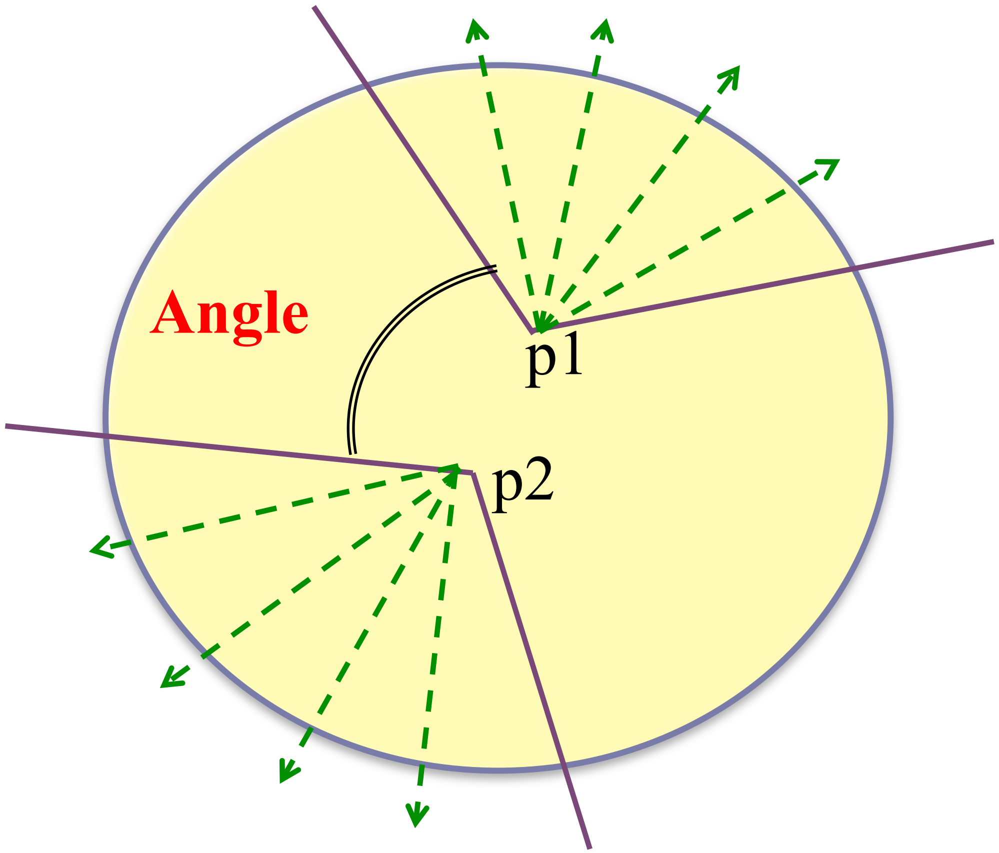


Ryzom (MMORPG - OpenSource)

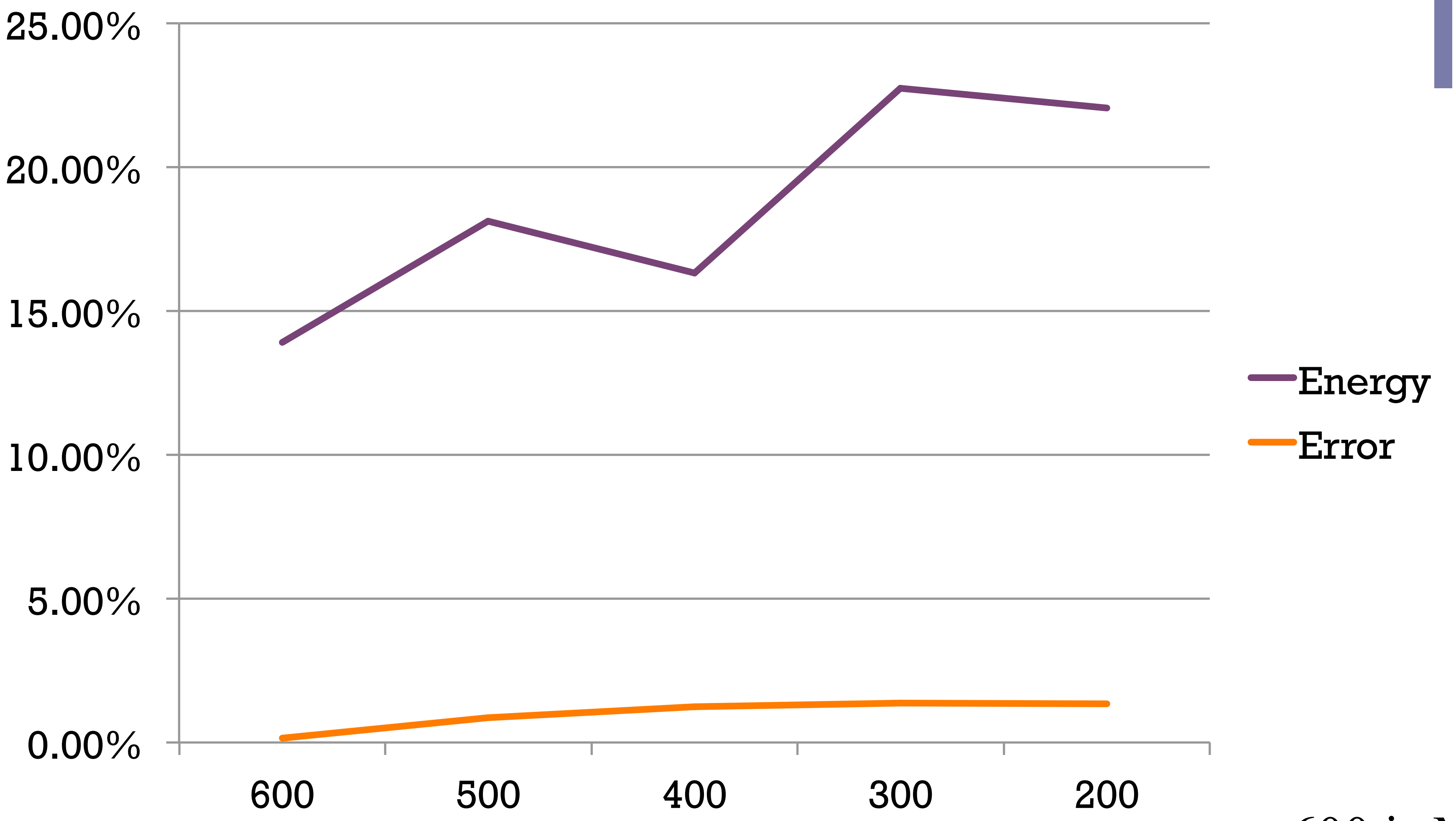
Implementation in Ryzom

- MACRO: Dual Ring Approach
  - With 200, 300, 400, 500 …. 1000ms time steps
- MICRO: Viewing Angle



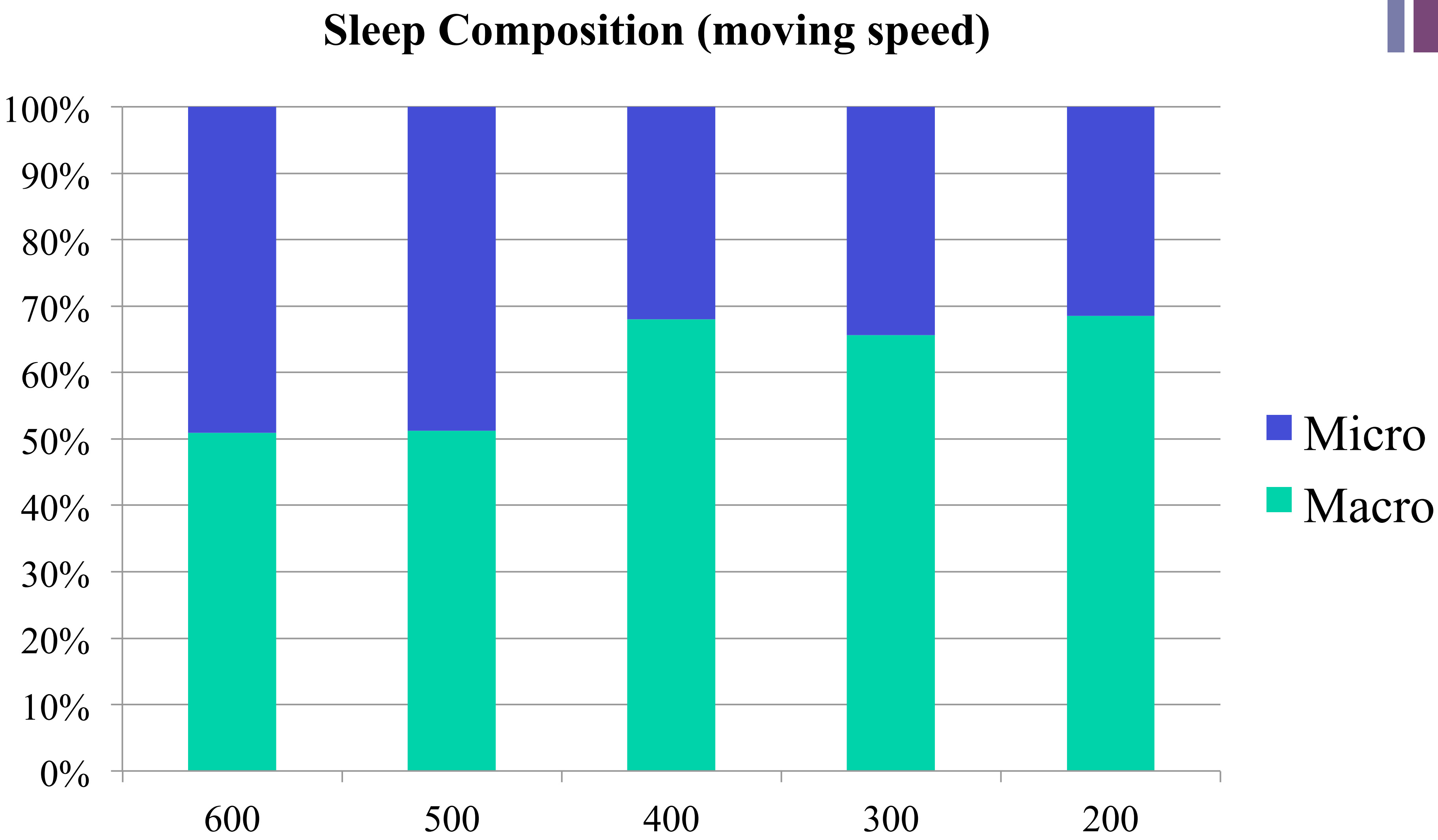


Results (for 40 players)



600 is Most conservative on Quality

Results (for 40 players)



Quake III (FPS)

Implementation in Quake III

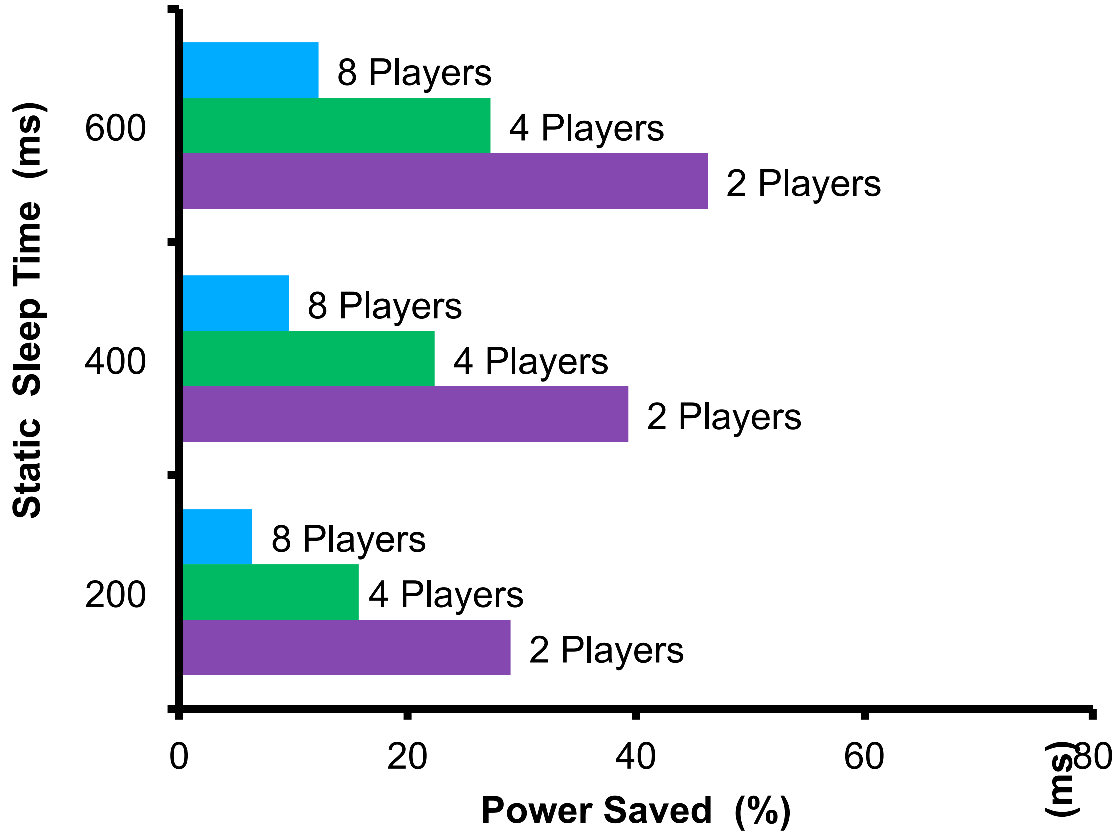
- Performance depends on map size and number of players
- On an average map, we are able to save up to **25% of network power** with little noticeable impact to the game
- **User study conducted to study impact**
- Any artifacts only manifest when the player first comes into vision

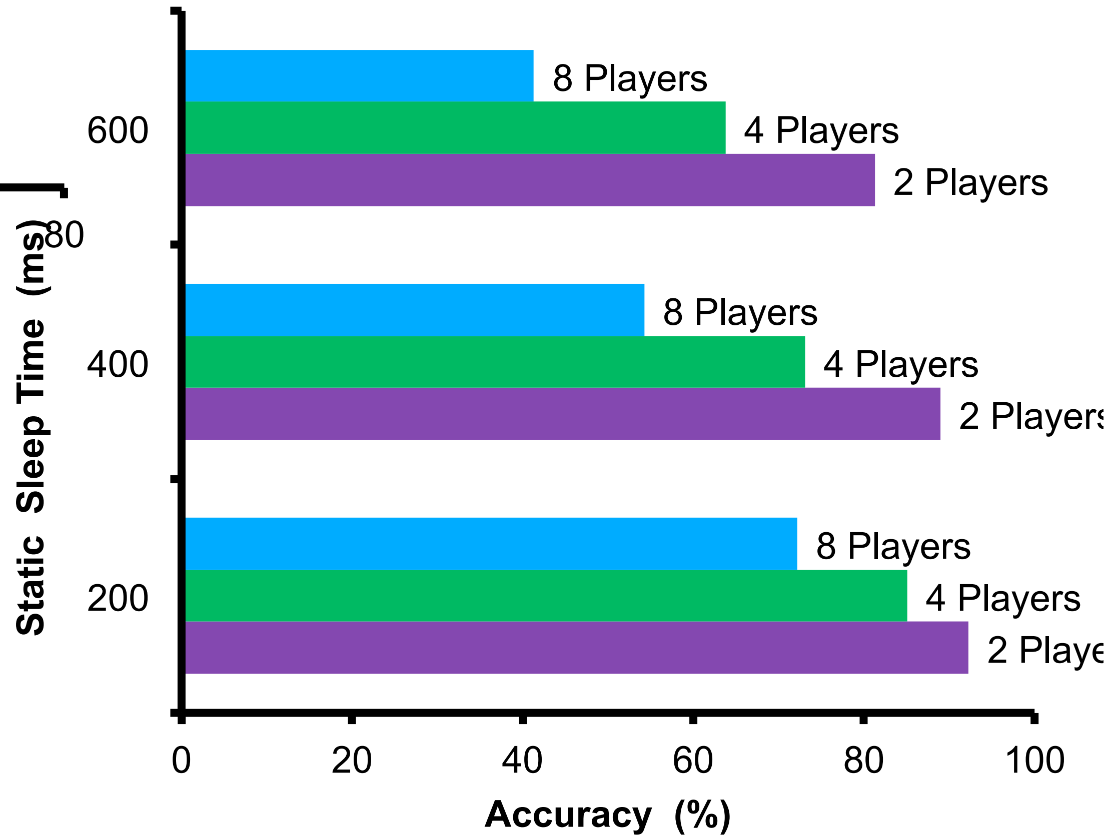
Implementation in Quake III

- Modes (Algorithm Variations):
- Static 200
- Static 400
- Static 600
- Dynamic

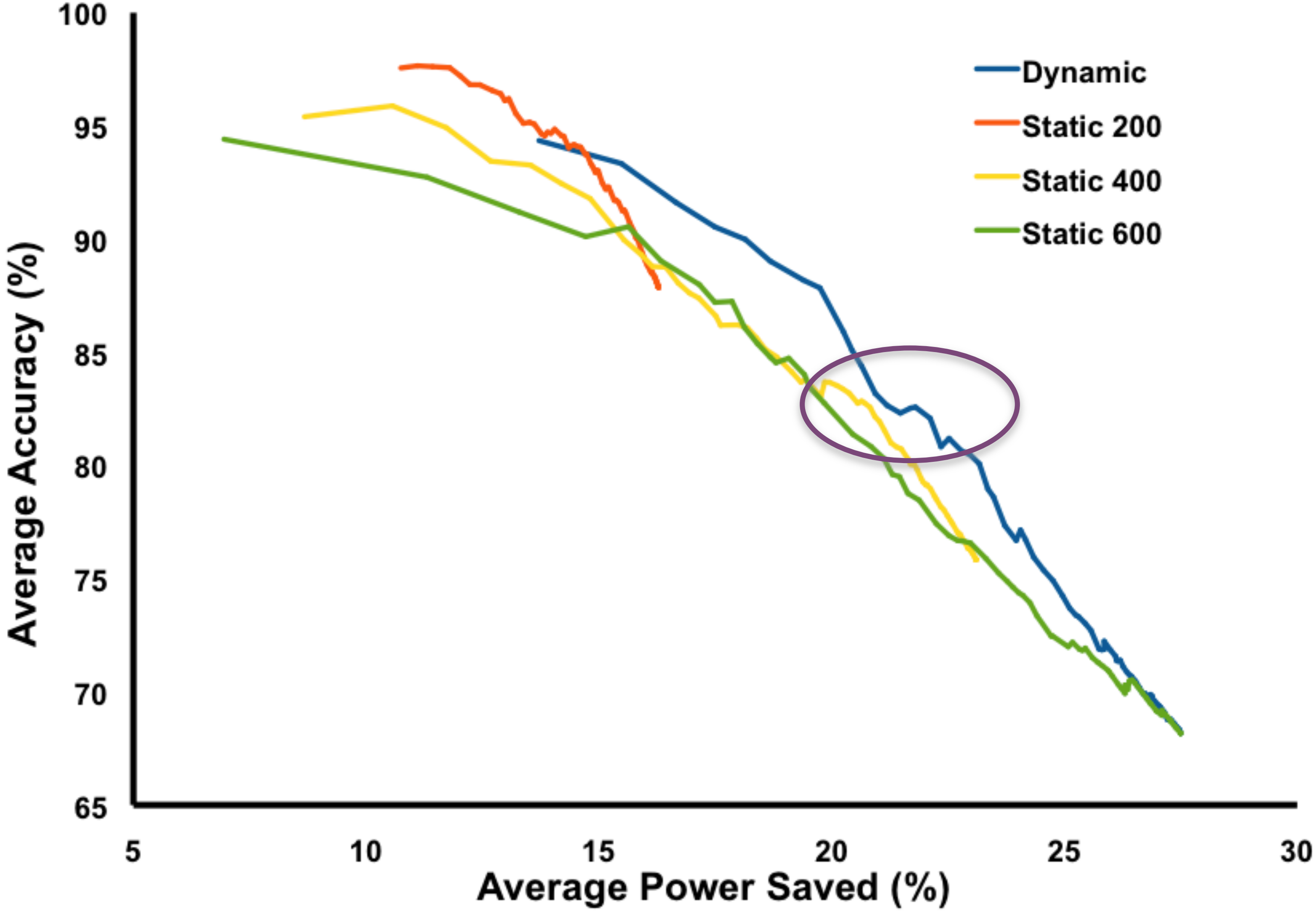
(-3,3)	(-2,3)	(-1,3)	(0,3)	(1,3)	(2,3)	(3,3)
(-3,2)	(-2,2)	(-1,2)	(0,2)	(1,2)	(2,2)	(3,2)
(-3,1)	(-2,1)	(-1,1)	(0,1)	(1,1)	(2,1)	(3,1)
(-3,0)	(-2,0)	(-1,0)	(0,0)	(1,0)	(2,0)	(3,0)
(-3,-1)	(-2,-1)	(-1,-1)	(0,-1)	(1,-1)	(2,-1)	(3,-1)
(-3,-2)	(-2,-2)	(-1,-2)	(0,-2)	(1,-2)	(2,-2)	(3,-2)
(-3,-3)	(-2,-3)	(-1,-3)	(0,-3)	(1,-3)	(2,-3)	(3,-3)

Baseline Results

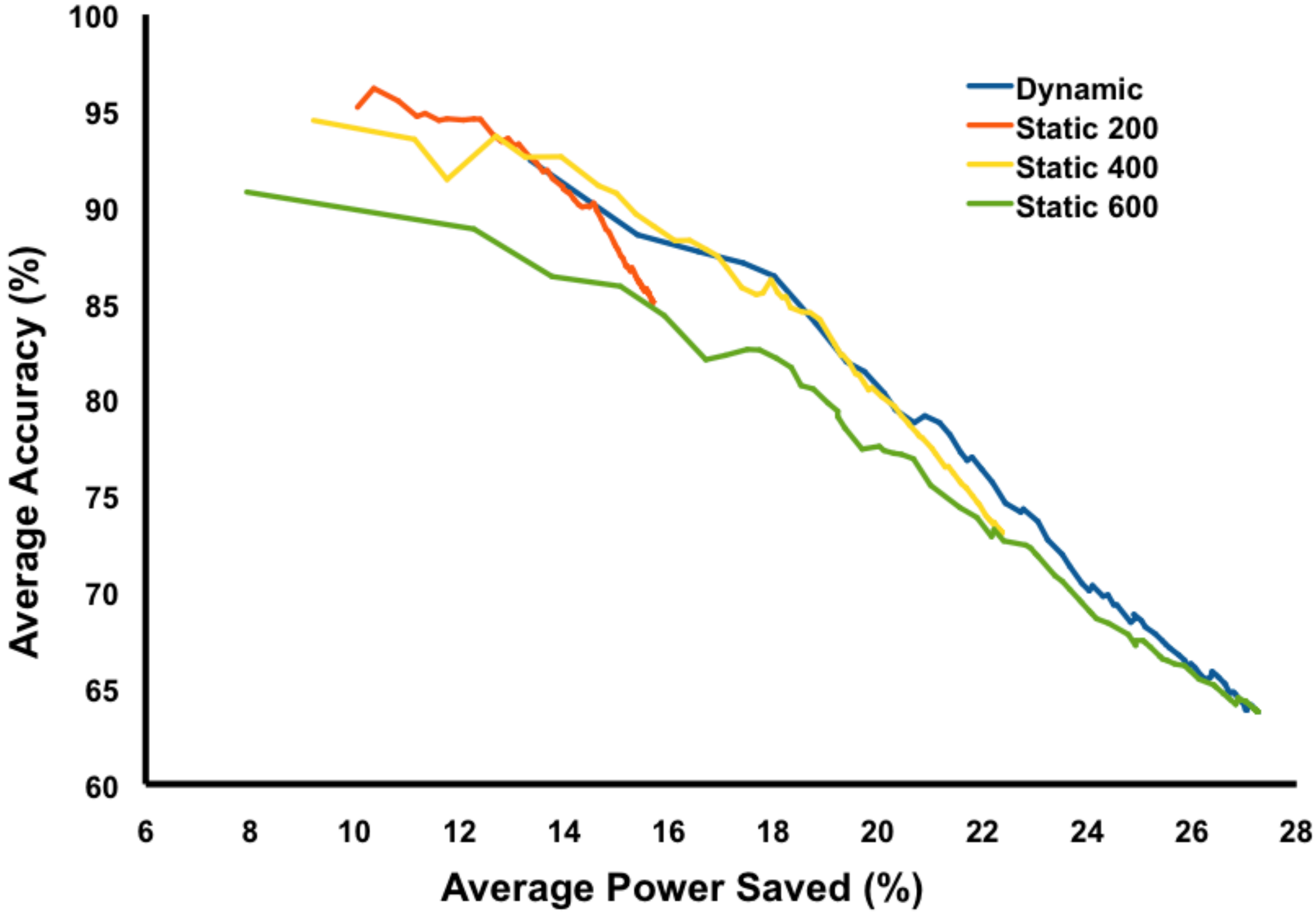




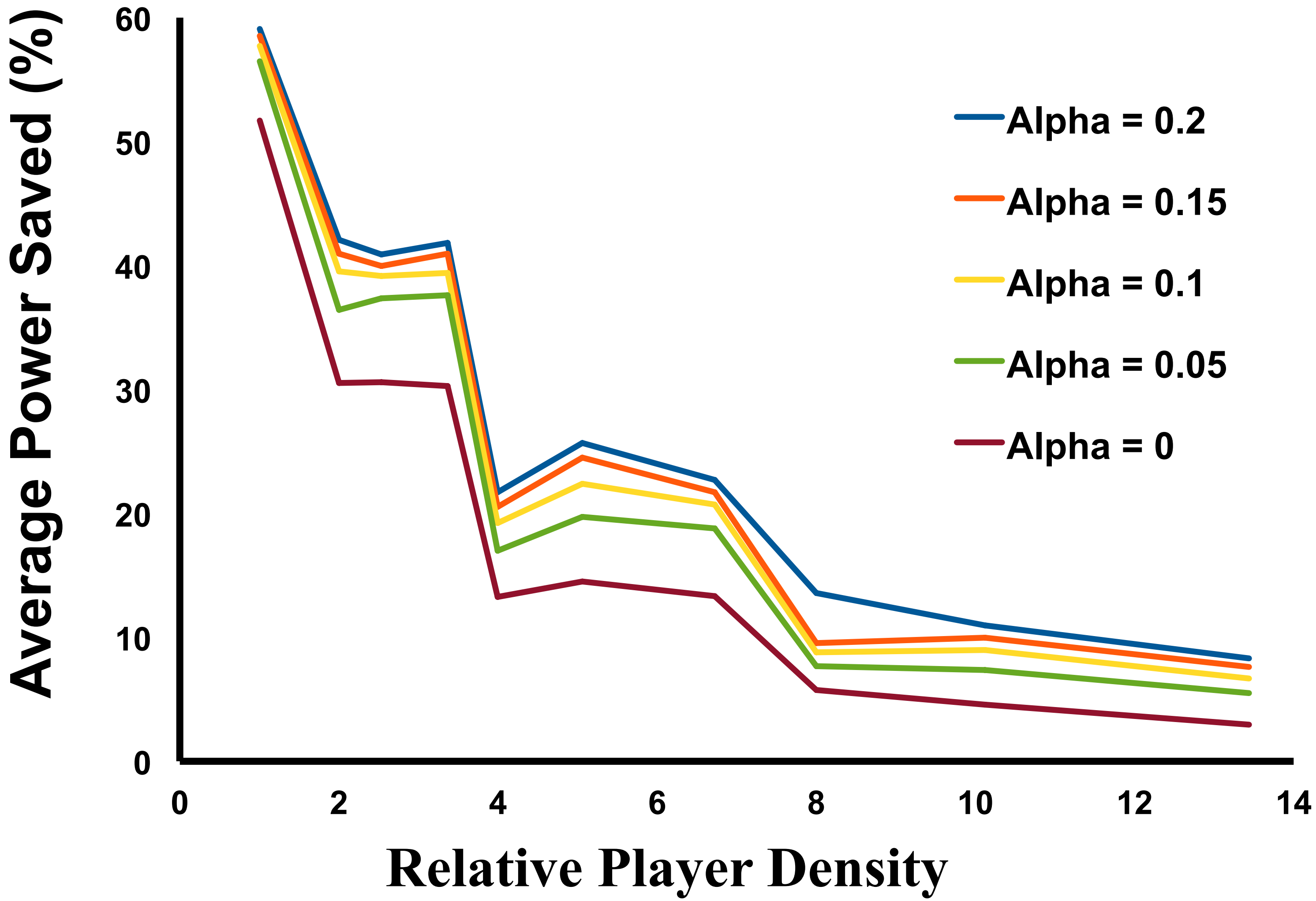
Results over 3G



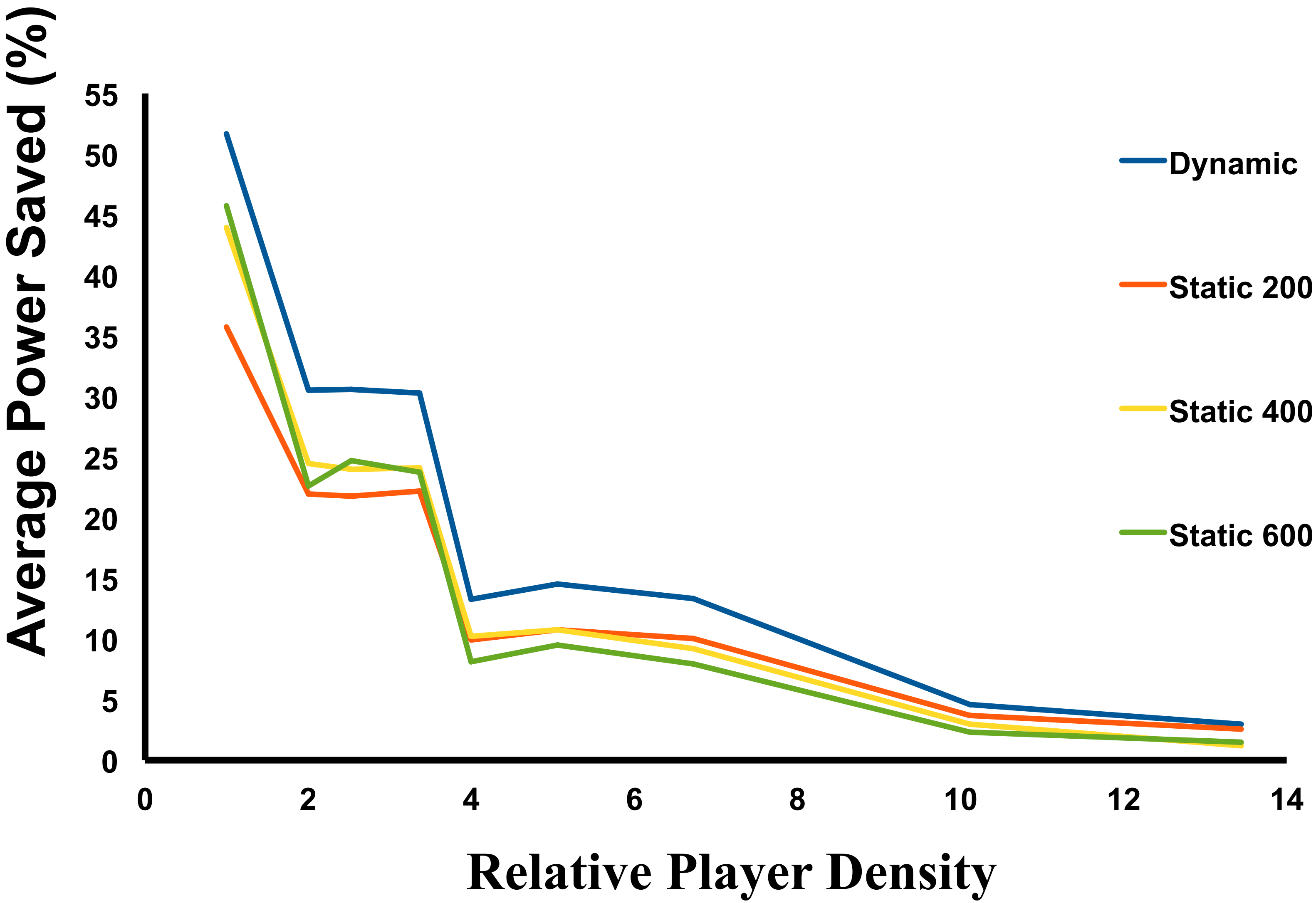
Results over Wifi



Effect Of Density & Dynamic Algorithm

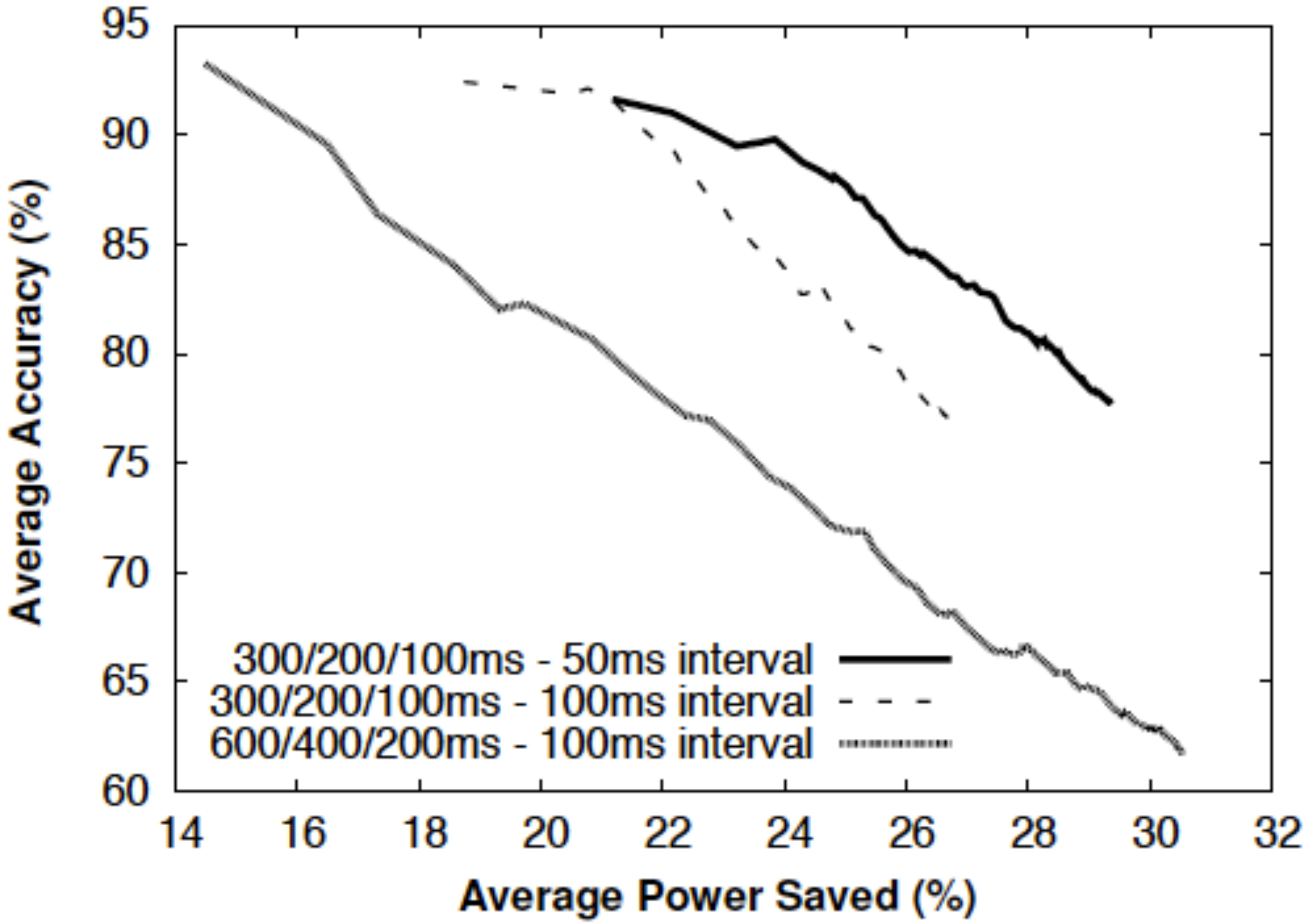


+ Effect Of Density & Dynamic Algorithm

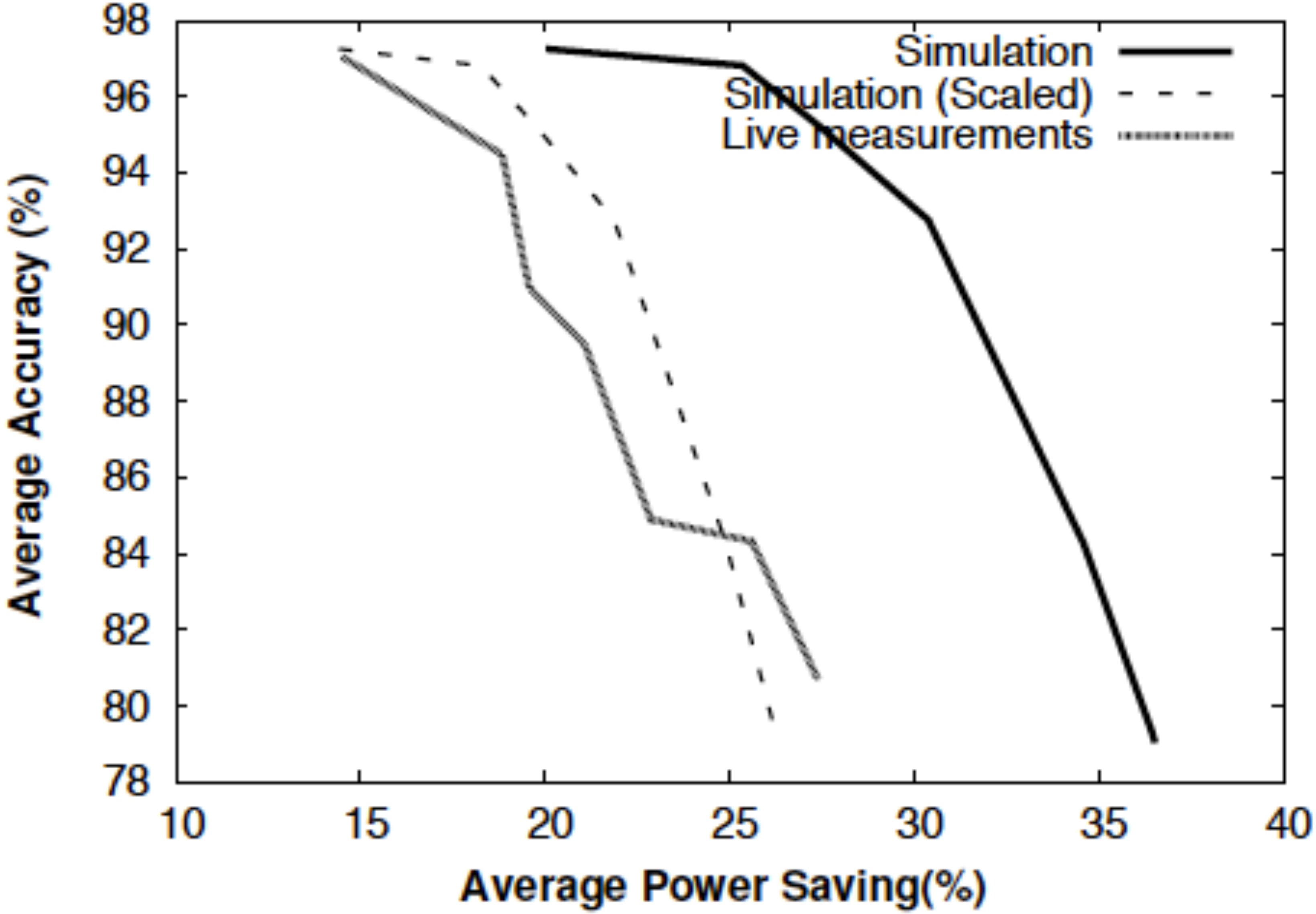


Results - Visibility

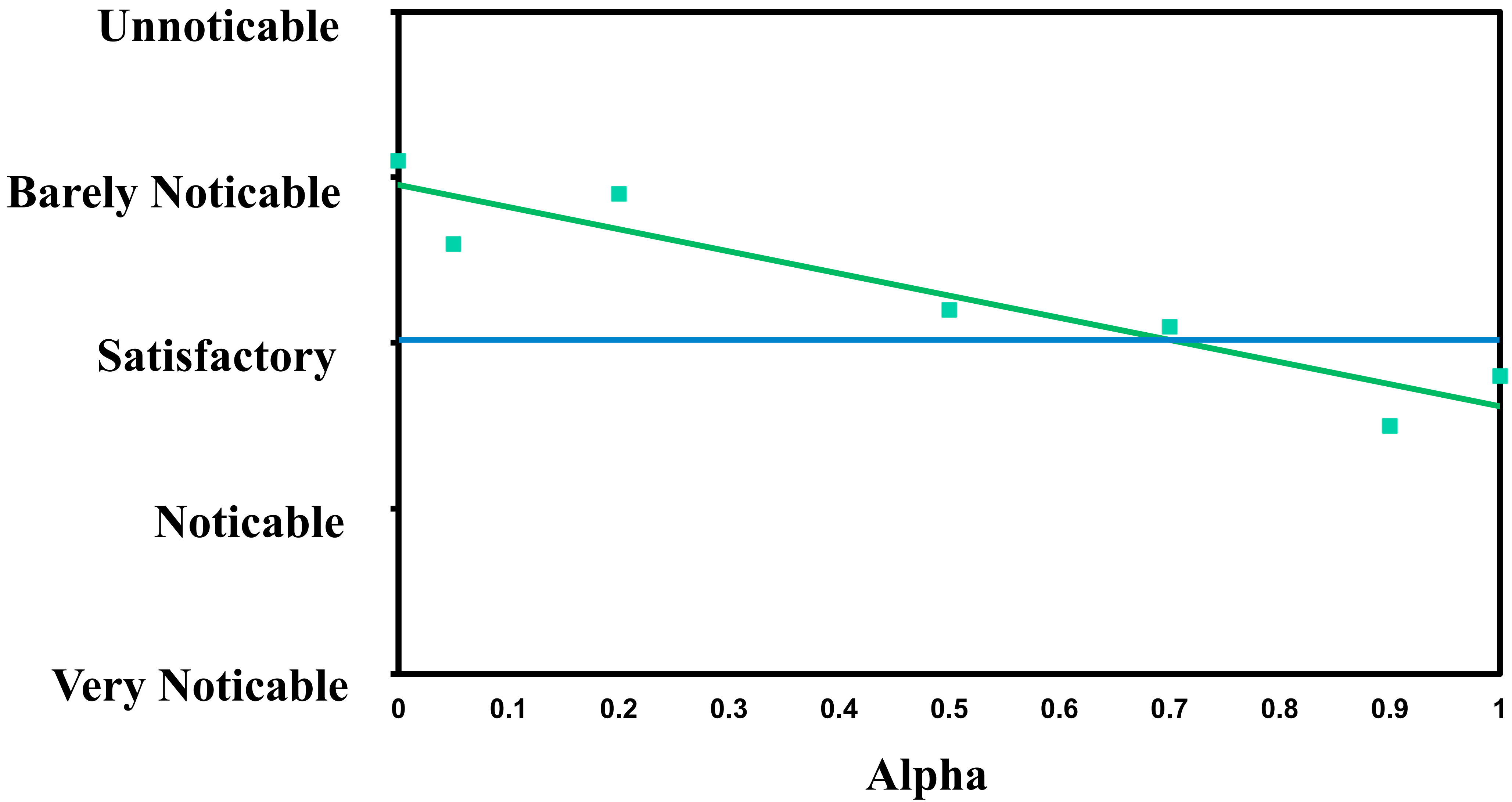
Effect of Different sleep intervals



Actual Measurements



User Study Results - Game Quality Impact



Intro to Renderer's View/PVS Based

Implementation in Quake III

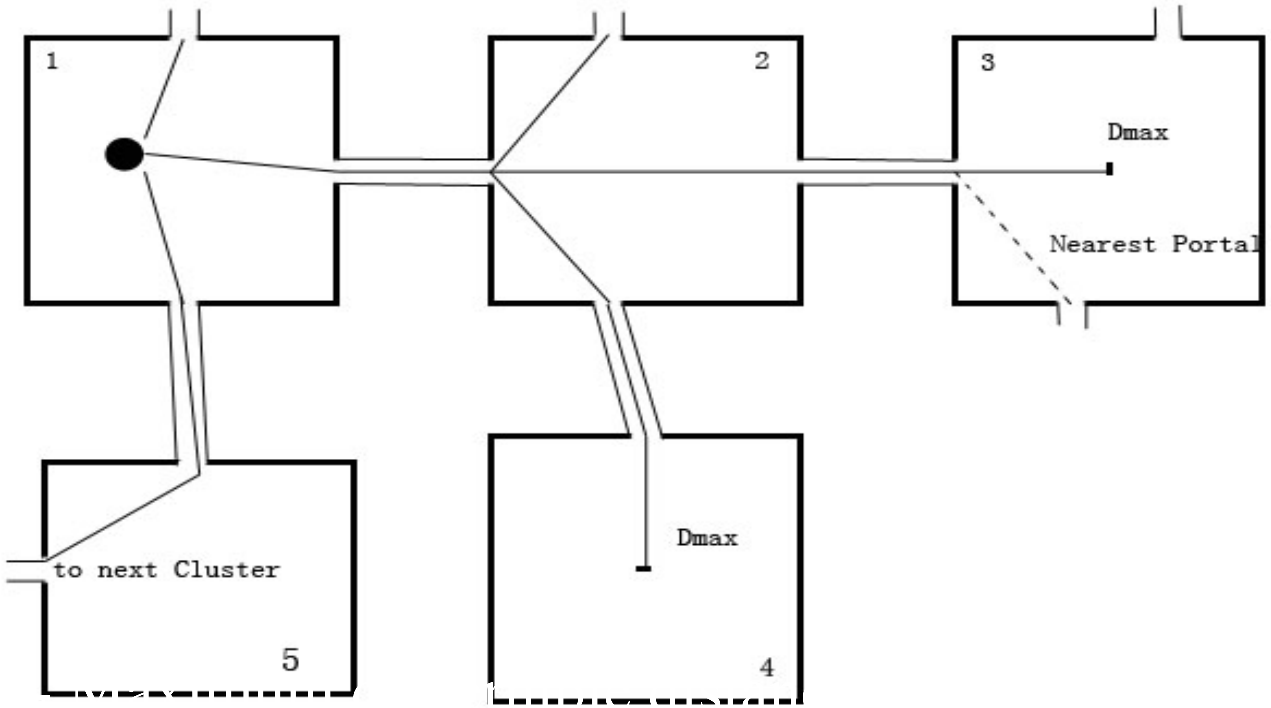
- MACRO – Single Ring (Dmax), Cluster Level, Path Distance
- MICRO – Area Level Visibility

Intro to Renderer's View/PVS Based

Implementation in Quake III

Macro Algorithm

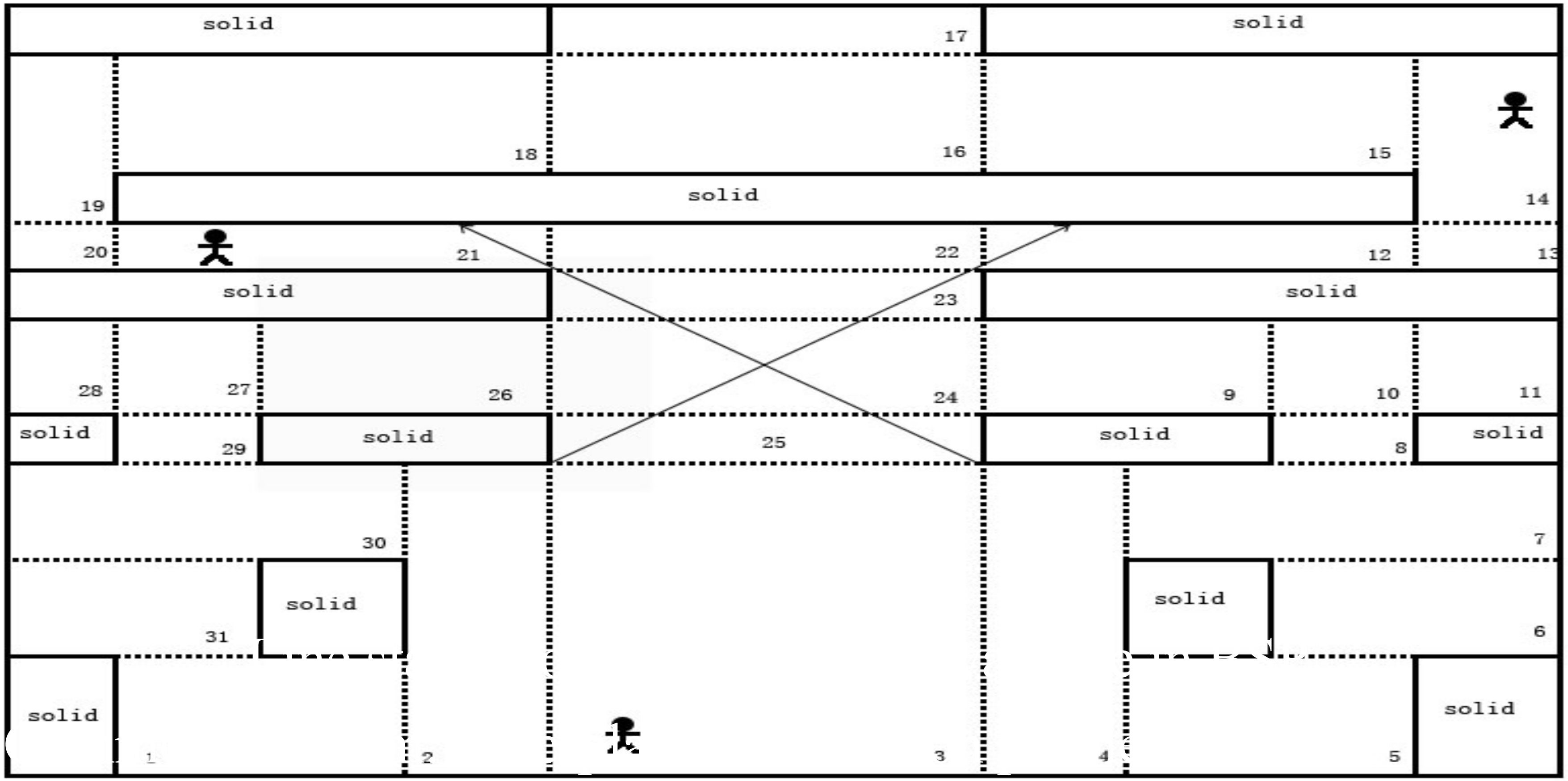
- A BFS is carried on the clusters, using the BSP tree



Intro to Renderer's View/PVS Based

Implementation in Quake III

- Micro Scanning - Pre-computes "Potentially Visible Set" and Stores in BSP tree



+

THE END

THANKS FOR YOUR ATTENTION

Questions?

Questions?

Questions?