A Modeling Framework for Optimizing the Flow-Level Scheduling with Time-Varying Channels

Peter Jacko*
joint work with Urtzi Ayesta and Martin Erausquin

Performance 2010, November 17

*BCAM — Basque Center for Applied Mathematics, Spain

Motivation: Wireless Downlink

- Channel conditions vary due to fading
- Exponential-length jobs
- Channel conditions independent across users
- i.i.d. channel conditions from slot to slot

- Base station can serve 1 user per slot

Talk Outline

- Resource allocation problem (restless bandit extension)
- MDP framework
- Threshold policies and indexability
- Potential improvement (index) rule
- Application in wireless networks
- Performance evaluation by simulations
- Work in progress

Resource Allocation Problem (RAP)

- Stochastic and dynamic
- There are a number of independent users
- Constraint: resource capacity at every moment
- Objective: maximize expected “reward”
- Captures the exploitation vs. exploration trade-off
  - ▷ always exploiting (being myopic) is not optimal
  - ▷ always exploring (being utopic) is not optimal
- This is a model of learning by doing!

Adaptive Greedy Rules

- Assign a dynamic price (index value) to each user
- We are concerned with the following rule
  - ▷ given the situation at each moment, be greedy: serve job with highest current price
- Experiments and simulations suggest that it gives a nearly-optimal solution to RAP
- In some problems it is optimal
  - ▷ $c\mu$-rule (Cox & Smith '61): job sequencing
  - ▷ Gittins index rule ('72): multi-armed bandit problem
  - ▷ Klimov index rule ('74): $M/G/1$ model w/ feedback

MDP Framework

- Markov Decision Processes
- Discrete time model $(t = 0, 1, 2, \dots)$
- Job $k \in \mathcal{K}$ is defined by
  - ▷ state space $\mathcal{N}_k$, action space $\mathcal{A}$
  - ▷ expected one-period capacity consumption $\boldsymbol{W}_k^a$
  - ▷ expected one-period reward $\boldsymbol{R}_k^a$
  - ▷ one-period transition probability matrix $\boldsymbol{P}_k^a$
- State process $X_k(t) \in \mathcal{N}_k$
- Action process $a_k(t) \in \mathcal{A}$ – to be decided

Time-Varying Job Sequencing Problem

- Job/user/channel $k \in \mathcal{K}$ is defined by
  - ▷ c_k = cost of waiting for job k
  - ▷ $q_{k,n}$ = probability to move to channel condition n (steady-state distribution)
  - ▷ $\mu_{k,n}$ = completion probability for job k under condition n (ordered: $\mu_{k,n} \leq \mu_{k,n+1}$)
- Find a serving sequence minimizing the total cost of waiting of jobs $k \in \mathcal{K}$
- $\mathcal{N}_k := \{0, 1, 2, \dots, N_k\}$, $\mathcal{A}_k := \{$'serve', 'wait'$\}$
- $0 =$ 'completed' ; $n =$ 'waiting' and condition is n

Time-Varying Job Sequencing Problem

- Expected one-period reward

$$R_{k,0}^{\text{`serve'}} := 0, \qquad R_{k,n}^{\text{`serve'}} := -c_k(1-\mu_{k,n}),$$
$$R_{k,0}^{\text{`wait'}} := 0, \qquad R_{k,n}^{\text{`wait'}} := -c_k;$$

- One-period transition probability matrices

$$\boldsymbol{P}_k^{\text{`serve'}} := \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ \vdots \\ N_k \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & \dots & N_k \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ \mu_{k,1} & \widetilde{\mu}_{k,1} q_{k,1} & \dots & \widetilde{\mu}_{k,1} q_{k,N_k} \\ \mu_{k,2} & \widetilde{\mu}_{k,2} q_{k,1} & \dots & \widetilde{\mu}_{k,2} q_{k,N_k} \\ \vdots & \vdots & \ddots & \vdots \\ \mu_{k,N_k} & \widetilde{\mu}_{k,N_k} q_{k,1} & \dots & \widetilde{\mu}_{k,N_k} q_{k,N_k} \end{pmatrix} \end{array}.$$

Resource Allocation Problem

- Formulation under the β-discounted criterion:

$$\max_{\pi \in \Pi} \sum_{k \in \mathcal{K}} \mathbb{E}^{\pi} \left[\sum_{t=0}^{\infty} \beta^t R_{k, X_k(t)}^{a_k(t)} \right]$$

subject to $\quad \sum_{k \in \mathcal{K}} W_{k, X_k(t)}^{a_k(t)} = W, \qquad$ for all $t = 0, 1, 2, \ldots$

- Analogously under the time-average criterion

- This problem is PSPACE-hard
  - ▷ intractable to solve exactly by Dynamic Programming
  - ▷ instead, we relax and decompose the problem

Whittle's Relaxation

- Serve W jobs in expectation
  - ▷ infinite number of constraints is replaced by one
  - ▷ sort of perfect market assumption

$$\max_{\pi\in\Pi} \sum_{k\in\mathcal{K}} \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \beta^t R_{k,X_k(t)}^{a_k(t)}\right]$$

$$\text{subject to} \quad \sum_{k\in\mathcal{K}} \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \beta^t W_{k,X_k(t)}^{a_k(t)}\right] = \sum_{t=0}^{\infty} \beta^t W$$

- Provides an upper bound for RAP

Lagrangian Relaxation

- Pay cost ν for using the server
  - ▷ the constraint is moved into the objective

$$\max_{\pi\in\Pi} \sum_{k\in\mathcal{K}} \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \beta^t R_{k,X_k(t)}^{a_k(t)}\right] - \nu \sum_{k\in\mathcal{K}} \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \beta^t W_{k,X_k(t)}^{a_k(t)}\right]$$

- Also provides an upper bound for RAP
- Decomposes due to user independence into single-user parametric subproblems
  - ▷ solved by identifying the efficiency frontier

Optimal Solution to Subproblems

- Theorem 1: Threshold policy is optimal
  - ▷ serve if the channel condition is above a threshold
  - ▷ wait if the channel condition is below a threshold
- Theorem 2: Problem is indexable, which implies
  - ▷ if $\nu \leq \nu_{k,n}^{\mathsf{PI}}$, then it is optimal to serve in the channel condition n
  - ▷ if $\nu \geq \nu_{k,n}^{\mathsf{PI}}$, then it it optimal to wait in the channel condition n
- $\nu_{k,n}^{\mathsf{PI}}$ is the dynamic price (index value)
- This gives rise to opportunistic policy

Potential Improvement Index

- Under discounted criterion:

$$\nu_{k,n}^{\mathsf{PI}} = \frac{c_k \mu_{k,n}}{(1-\beta) + \beta \sum_{m>n} q_{k,m}(\mu_{k,m} - \mu_{k,n})}$$

- Under time-average criterion:

$$\nu_{k,n}^{\mathsf{PI}} = \frac{c_k \mu_{k,n}}{\sum_{m>n} q_{k,m}(\mu_{k,m} - \mu_{k,n})} \text{ for } n \neq N_k, \qquad \nu_{k,N_k}^{\mathsf{PI}} = \infty$$

  - ▷ tie-breaking if in the best state: $c_k \mu_{k,N_k}$

- Rule: serve the job with highest actual PI index

Wireless Data Network

- CDMA 1xEV-DO: Slot duration $t_c = 1.67ms$
- Let job length B_k be exponentially distributed
  - ▷ Probability of departure if served Δ bits in slot is $\mathbb{P}\left[b \leq B_k \leq b + \Delta \middle| B_k > b\right] \approx \Delta / \mathbb{E}[B_k]$
- Let $s_{k,n}$ be service rate (bps) in condition n, then

$$\mu_{k,n} :\approx \frac{s_{k,n} \cdot t_c}{\mathbb{E}[B_k]}$$

- PI rule is independent of $\mathbb{E}[B_k]$
  - ▷ only tie-breaking becomes: $c_k s_{k,N_k} / \mathbb{E}[B_k]$

Other Scheduling Disciplines

- Relatively Best (Qualcomm CDMA standard, 2000):

$$\nu_{k,n}^{\mathsf{RB}} := \frac{\mu_{k,n}}{\sum_{m=1}^{N_k} q_{k,m}\mu_{k,m}}$$

  ▷ ≈ Proportionally Fair scheduler (Borst, 2005)

- Score Based (Bonald, 2004): $\nu_{k,n}^{\mathsf{SB}} := \sum_{m=1}^{n} q_{k,m}$

- Proportionally Best: $\nu_{k,n}^{\mathsf{PB}} = \frac{\mu_{k,n}}{\mu_{k,N_k}}$

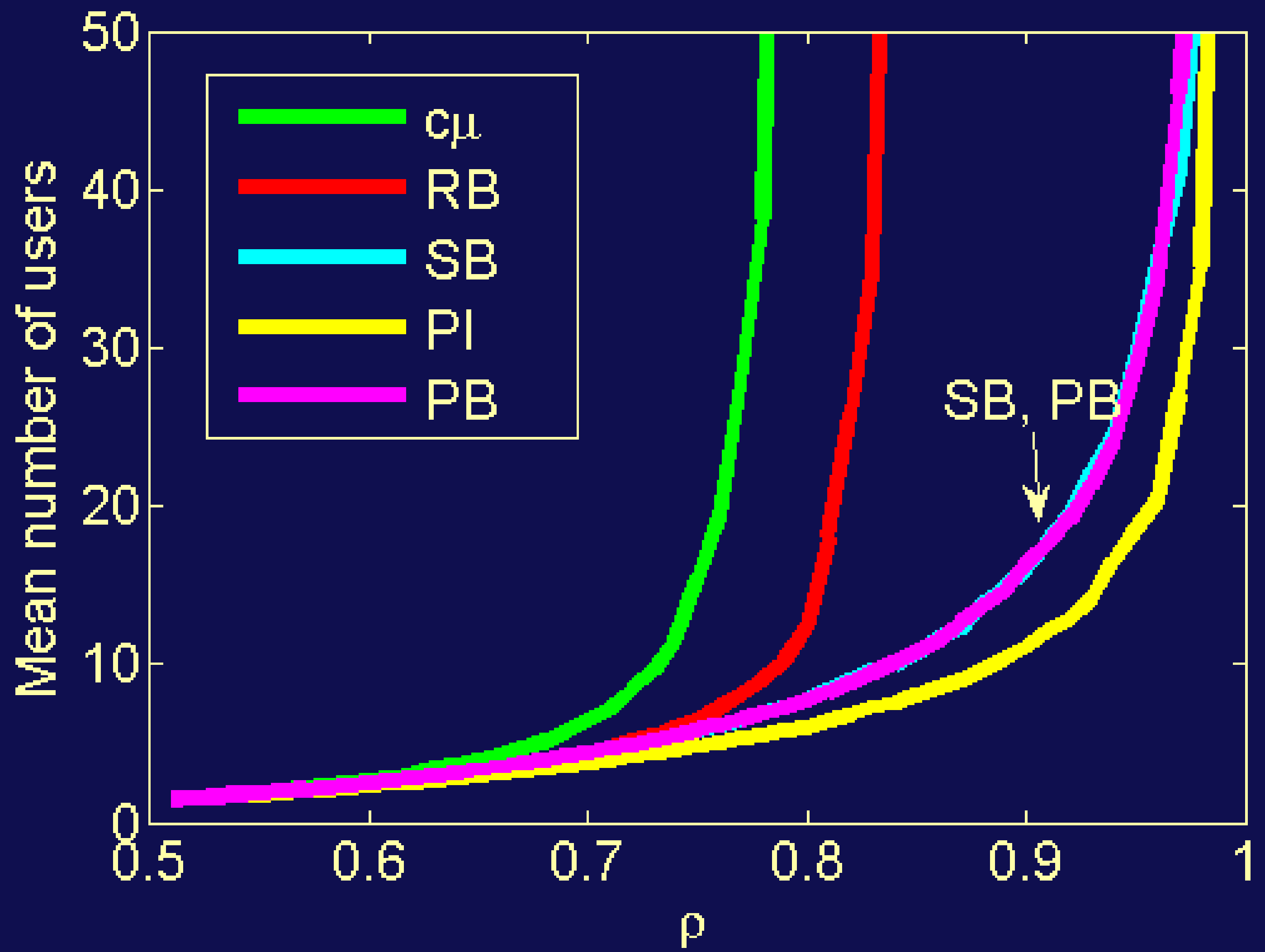
  ▷ maximum stability region (Aalto & Lassila, 2010)

Systems with Random Arrivals

- PI rule has maximum stability region
  - ▷ the only rule under general c_k's
- PI equivalent to RB in "symmetric" systems
  - ▷ performance characterized as processor sharing
- We evaluate performance in simulations
  - ▷ consider 2 different classes of jobs
  - ▷ λ_k: probability of arrival from class k

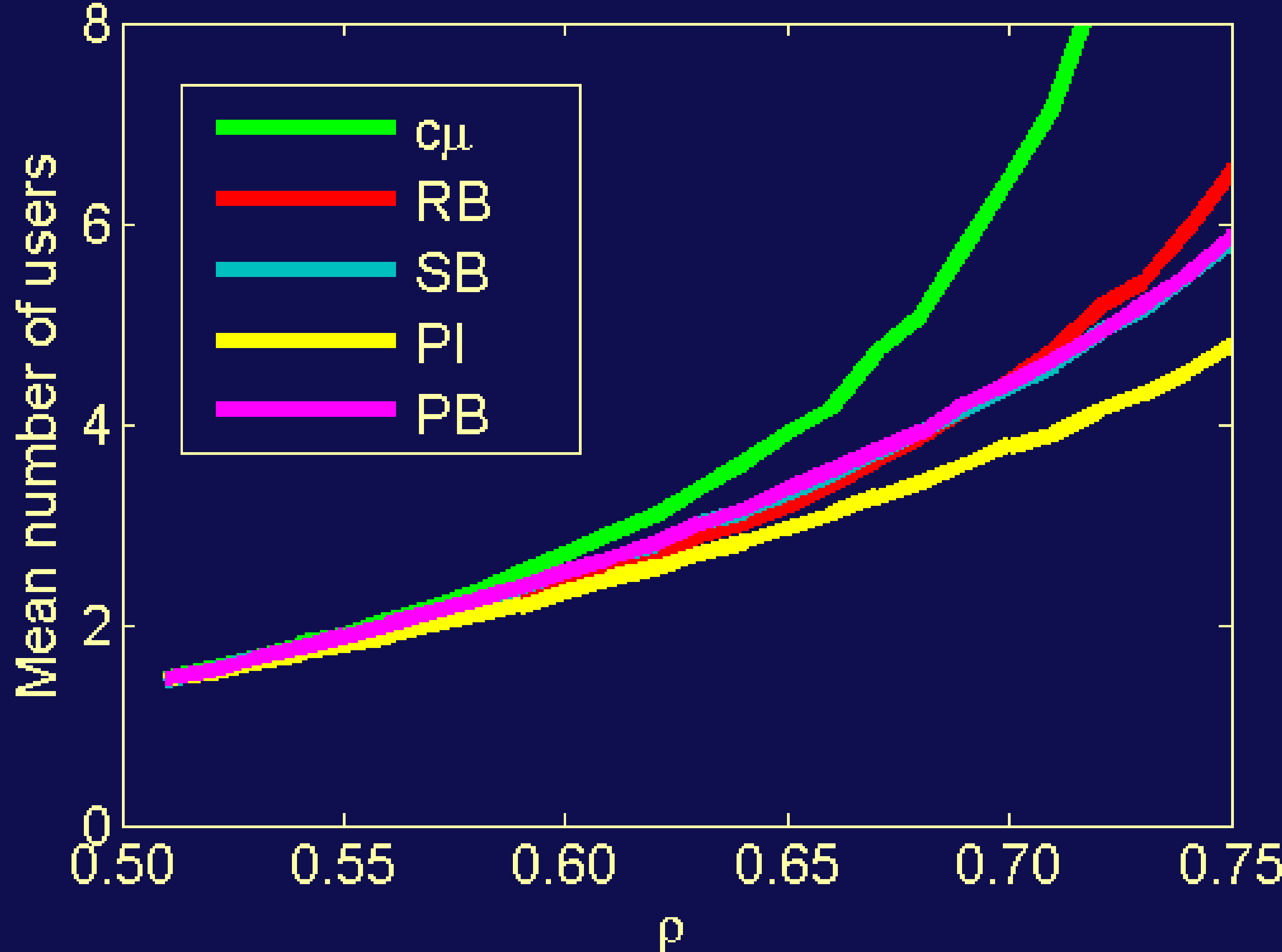
Numerical Simulations: Scenario 1

- Varied λ_1 so that ϱ varies from 0.5 to 1



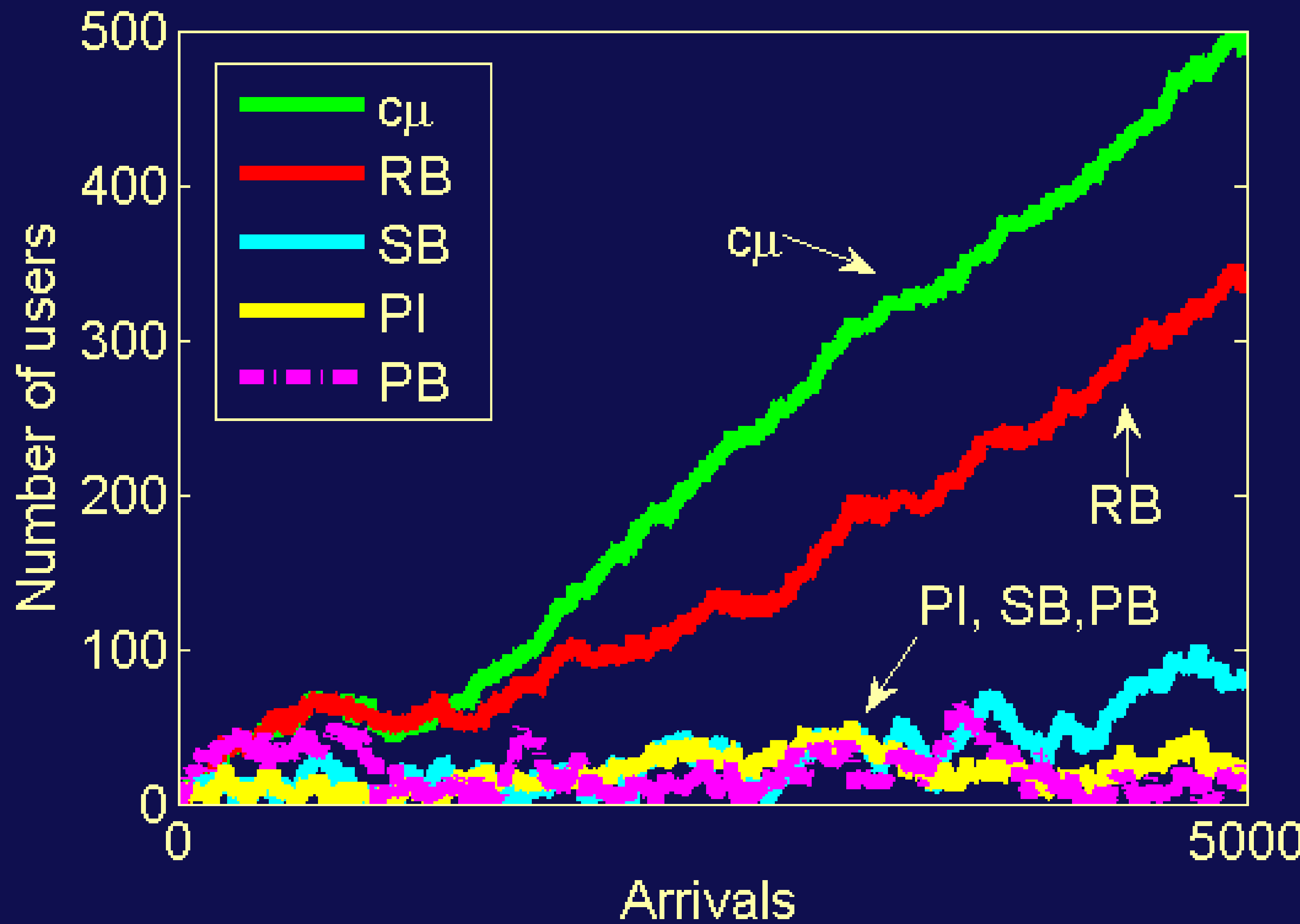
Numerical Simulations: Scenario 1

- Varied λ_1 so that ϱ varies from 0.5 to 1



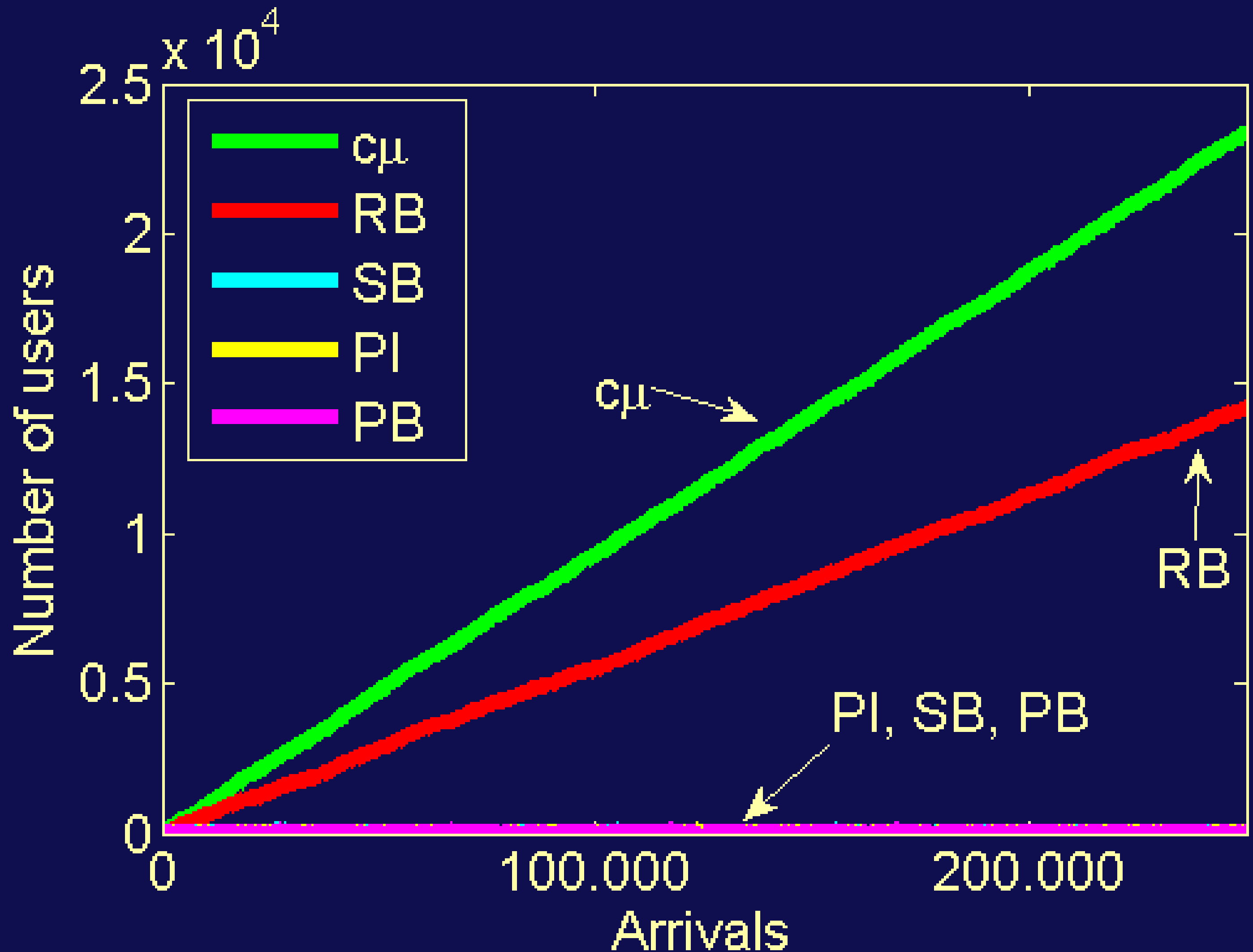
Numerical Simulations: Scenario 1

- Sample path of the number of users, $\varrho = 0.95$



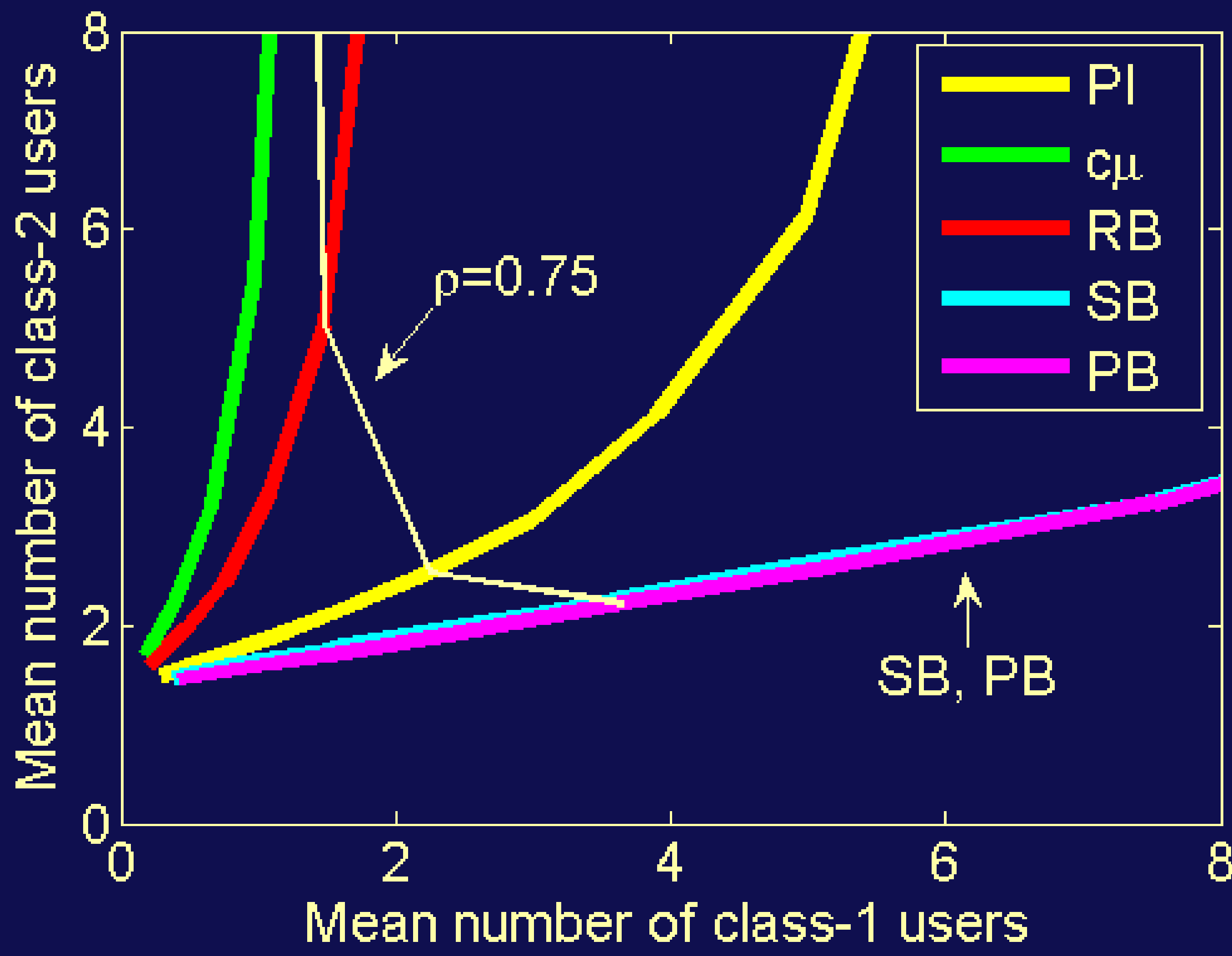
Numerical Simulations: Scenario 1

- Sample path of the number of users, $\varrho = 0.95$



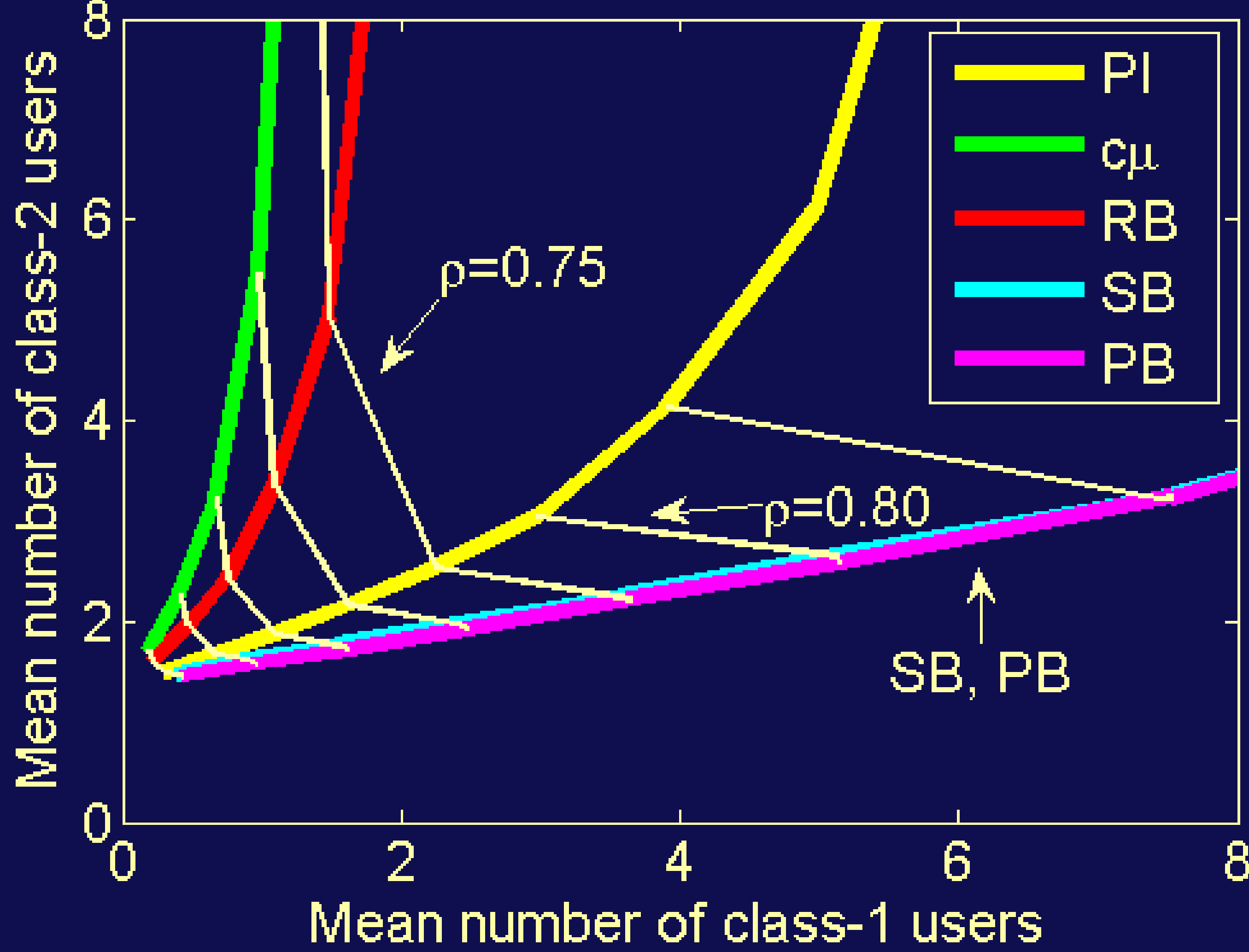
Numerical Simulations: Scenario 1

- Indifference curves for mean number of users



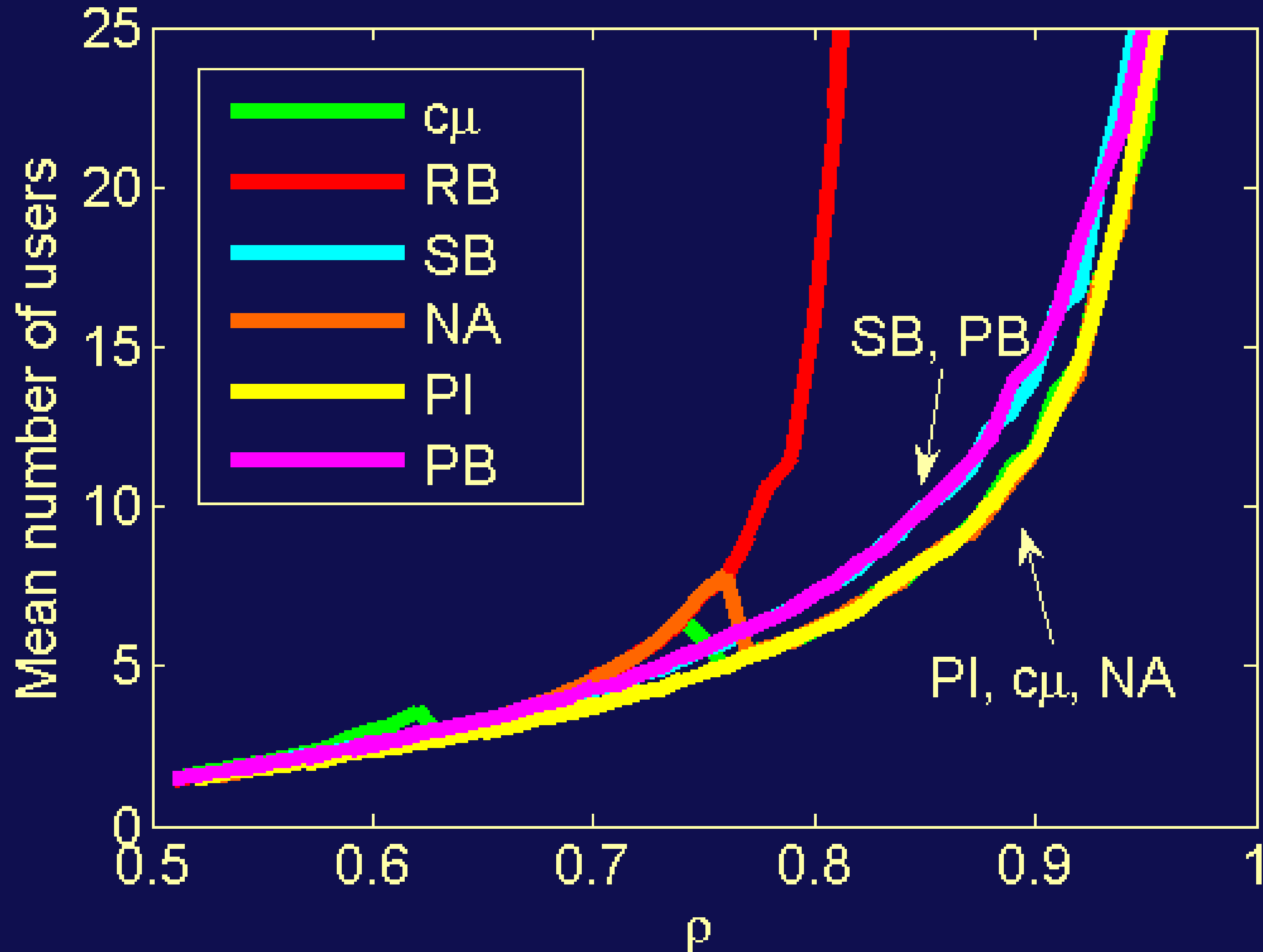
Numerical Simulations: Scenario 1

- Indifference curves for mean number of users



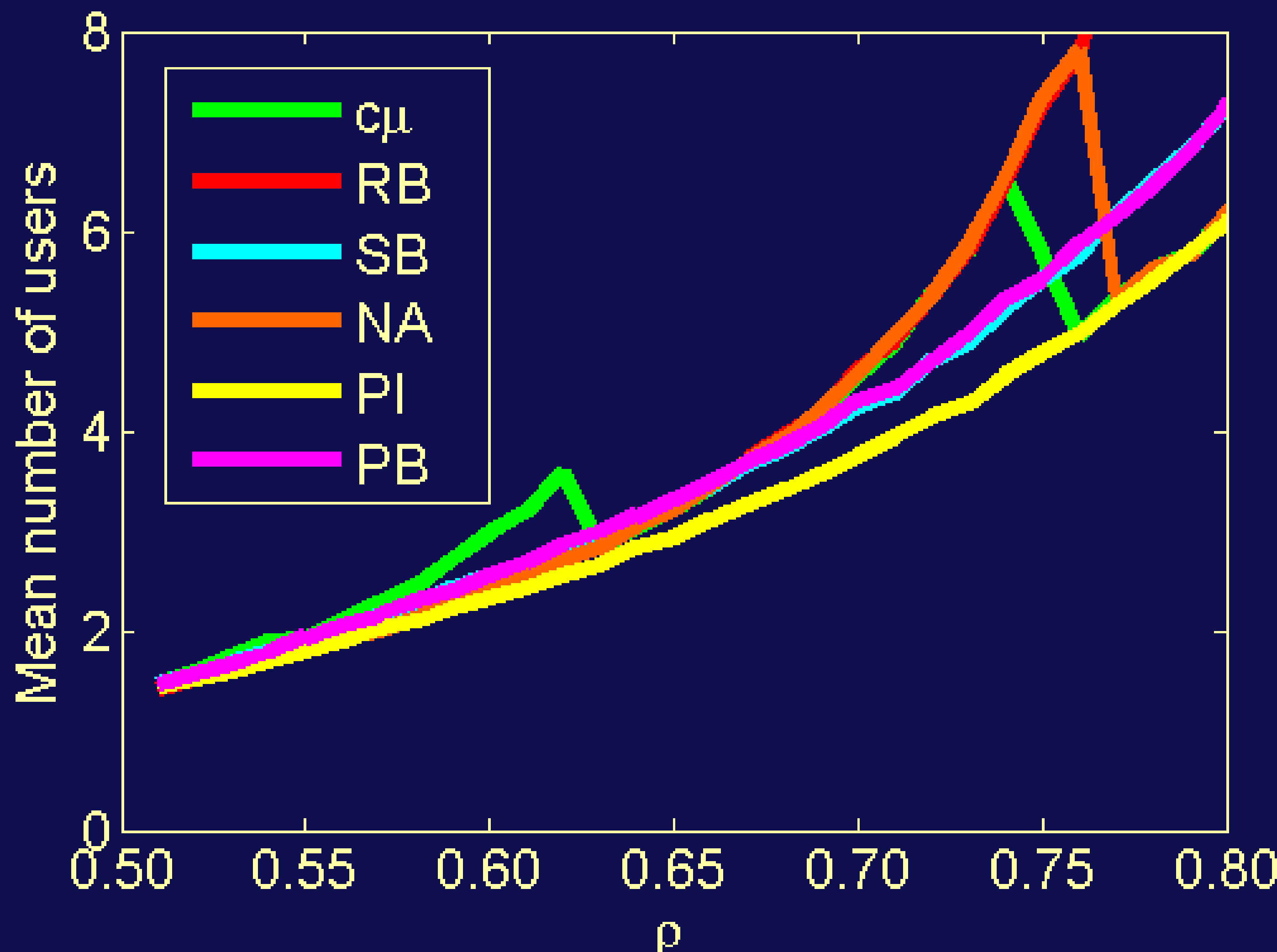
Numerical Simulations: Scenario 2

- Varied class-1 job length so that ϱ varies from 0.5 to 1



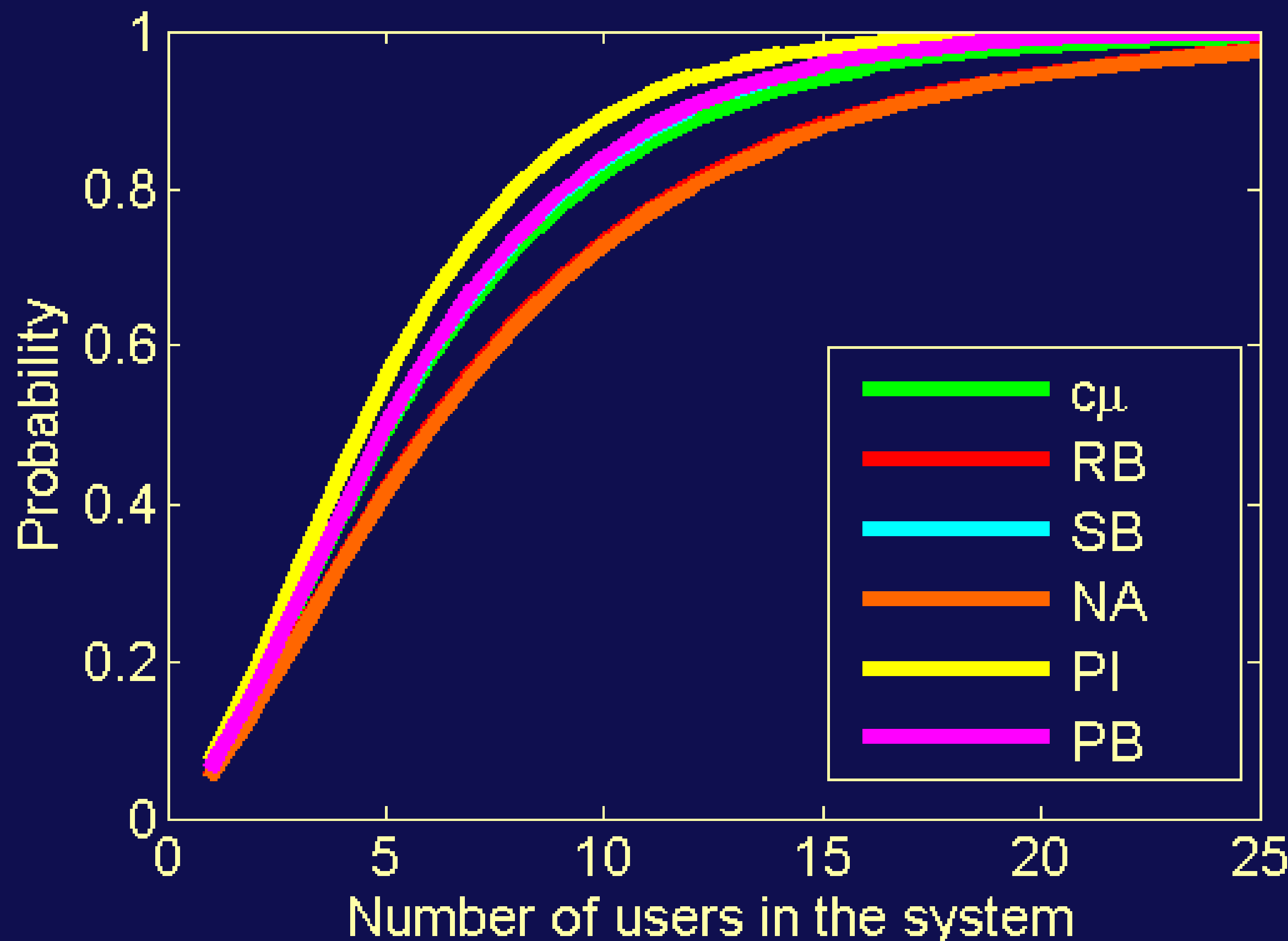
Numerical Simulations: Scenario 2

- Varied class-1 job length so that ϱ varies from 0.5 to 1



Numerical Simulations: Stoch. Dominance

- Typical picture of empirical CDFs



Simulations Summary

- PI consistently outperforms all the other rules
- Or its mean performance is equivalent to the best one
- Simulations strongly suggest stochastic dominance of PI over the other rules
- The stability region is the maximum for PI rule, while it is not for $c\mu$ and RB rules

Conclusion

- Framework to study opportunistic policies
  - ▷ RB (PF), PB roughly recovered under other rewards
- Tractable framework to obtain a new PI policy
  - ▷ asymptotically fluid-optimal (AEJV '10)
  - ▷ the only maximally stable policy in general (AL '10)
  - ▷ excellent performance in small-scale problems
- PI policy implies (roughly):
  - ▷ in low load: be channel-opportunistic
  - ▷ in high load: take into account job size ($c\mu$)

Future Research

- Work in progress
  - ▷ heavy-traffic/overload analysis of PI
  - ▷ PI with abandonments
  - ▷ non-iid channel evolution (fading or mobility)
- Open problems
  - ▷ optimal solution (structure)
  - ▷ online learning of PI parameters
  - ▷ theoretical justification of second-order index
  - ▷ correlation among users' channels

Thank you for your attention

Example: Job Sequencing Problem

- Find a serving sequence minimizing the total cost of waiting of jobs $k \in \mathcal{K}$
  - ▷ c_k = cost of waiting for job k
  - ▷ μ_k = completion probability for job k
- $\mathcal{N}_k := \{\text{'completed'}, \text{'waiting'}\}$, $\mathcal{A}_k := \{\text{'serve'}, \text{'wait'}\}$
- Expected one-period capacity consumption

$$W_{k,\text{'completed'}}^{\text{'serve'}} := 1, \qquad W_{k,\text{'waiting'}}^{\text{'serve'}} := 1,$$
$$W_{k,\text{'completed'}}^{\text{'wait'}} := 0, \qquad W_{k,\text{'waiting'}}^{\text{'wait'}} := 0;$$

Example: Job Sequencing Problem

- Expected one-period reward

$$R_{k,\text{‘completed’}}^{\text{‘serve’}} := 0, \qquad R_{k,\text{‘waiting’}}^{\text{‘serve’}} := -c_k(1-\mu_k),$$
$$R_{k,\text{‘completed’}}^{\text{‘wait’}} := 0, \qquad R_{k,\text{‘waiting’}}^{\text{‘wait’}} := -c_k;$$

- One-period transition probability matrices

$$\boldsymbol{P}_k^{\text{‘serve’}} := \begin{array}{r} \\ \text{‘completed’} \\ \text{‘waiting’} \end{array} \begin{array}{c} \begin{array}{cc} \text{‘completed’} & \text{‘waiting’} \end{array} \\ \begin{pmatrix} 1 & 0 \\ \mu_k & 1-\mu_k \end{pmatrix} \end{array}$$

$$\boldsymbol{P}_k^{\text{‘wait’}} := \begin{array}{r} \\ \text{‘completed’} \\ \text{‘waiting’} \end{array} \begin{array}{c} \begin{array}{cc} \text{‘completed’} & \text{‘waiting’} \end{array} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{array}$$

Dynamic Prices (Index Values)

- We will assign a dynamic price to each user
- Arises in the solution of the parametric subproblem
  - ▷ optimal policy: use server iff price greater than ν
- Prices are values of ν when optimal solution changes
- However, such prices may not exist!
  - ▷ indexability has to be proved
- Price computation (if they exist):
  - ▷ in general, by parametric simplex method
  - ▷ by analysis sometimes obtained in a closed form

Optimal Solution to Subproblems

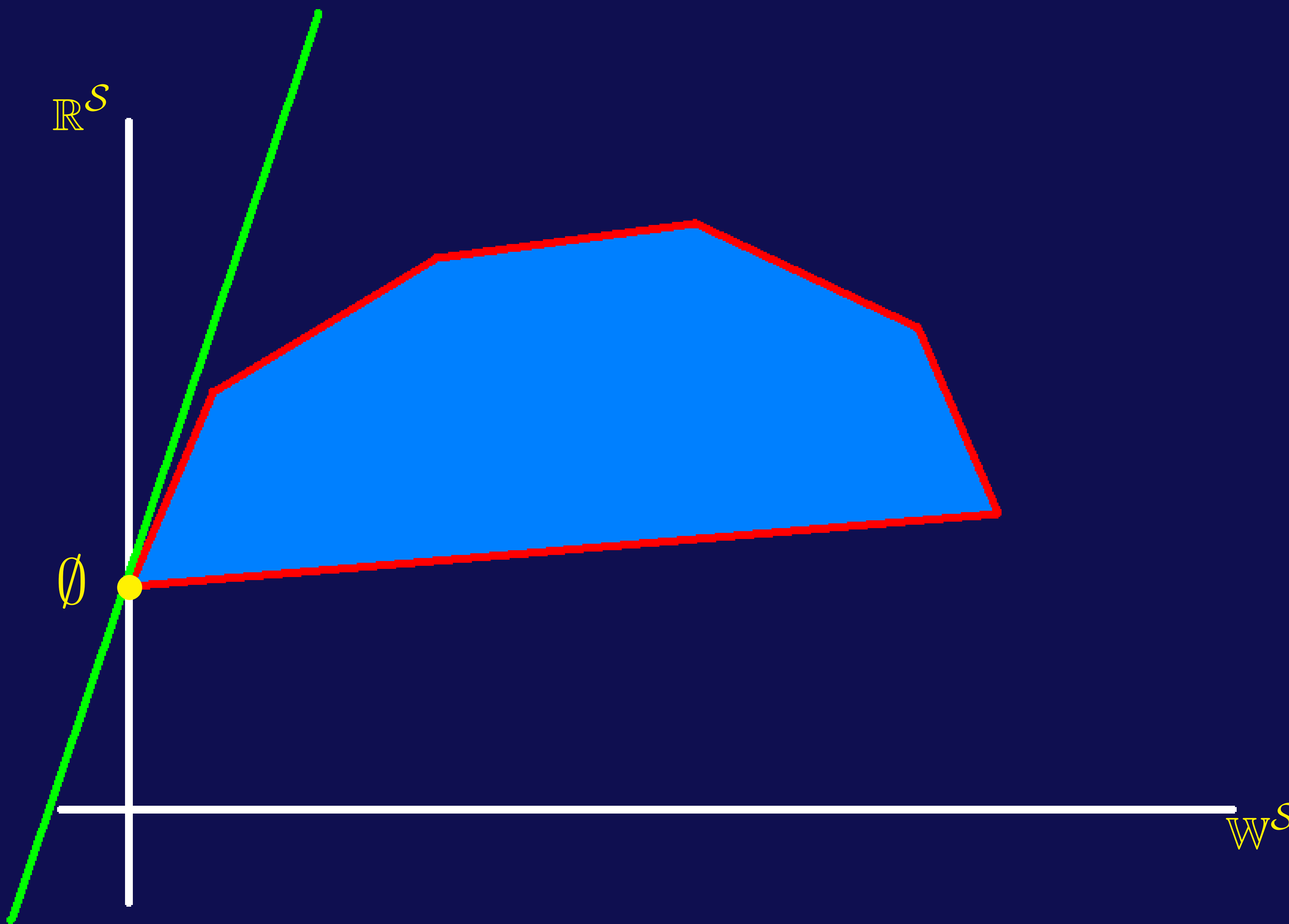
- For finite-state finite-action MDPs there exists an optimal policy that is deterministic, stationary, and independent of the initial state
  - ▷ we narrow our focus to those policies
  - ▷ represent them via serving sets $\mathcal{S} \subseteq \mathcal{N}$
  - ▷ policy $\mathcal{S}$ prescribes to serve in states in $\mathcal{S}$ and wait in states in $\mathcal{S}^{\mathsf{C}} := \mathcal{N} \setminus \mathcal{S}$
- Combinatorial ν-cost problem: $\max\limits_{\mathcal{S}\subseteq\mathcal{N}} \mathbb{R}_n^{\mathcal{S}} - \nu \mathbb{W}_n^{\mathcal{S}}$, where

$$\mathbb{R}_n^{\mathcal{S}} := \mathbb{E}_n^{\mathcal{S}} \left[\sum_{t=0}^{\infty} \beta^t R_{X(t)}^{a(t)} \right], \quad \mathbb{W}_n^{\mathcal{S}} := \mathbb{E}_n^{\mathcal{S}} \left[\sum_{t=0}^{\infty} \beta^t W_{X(t)}^{a(t)} \right]$$

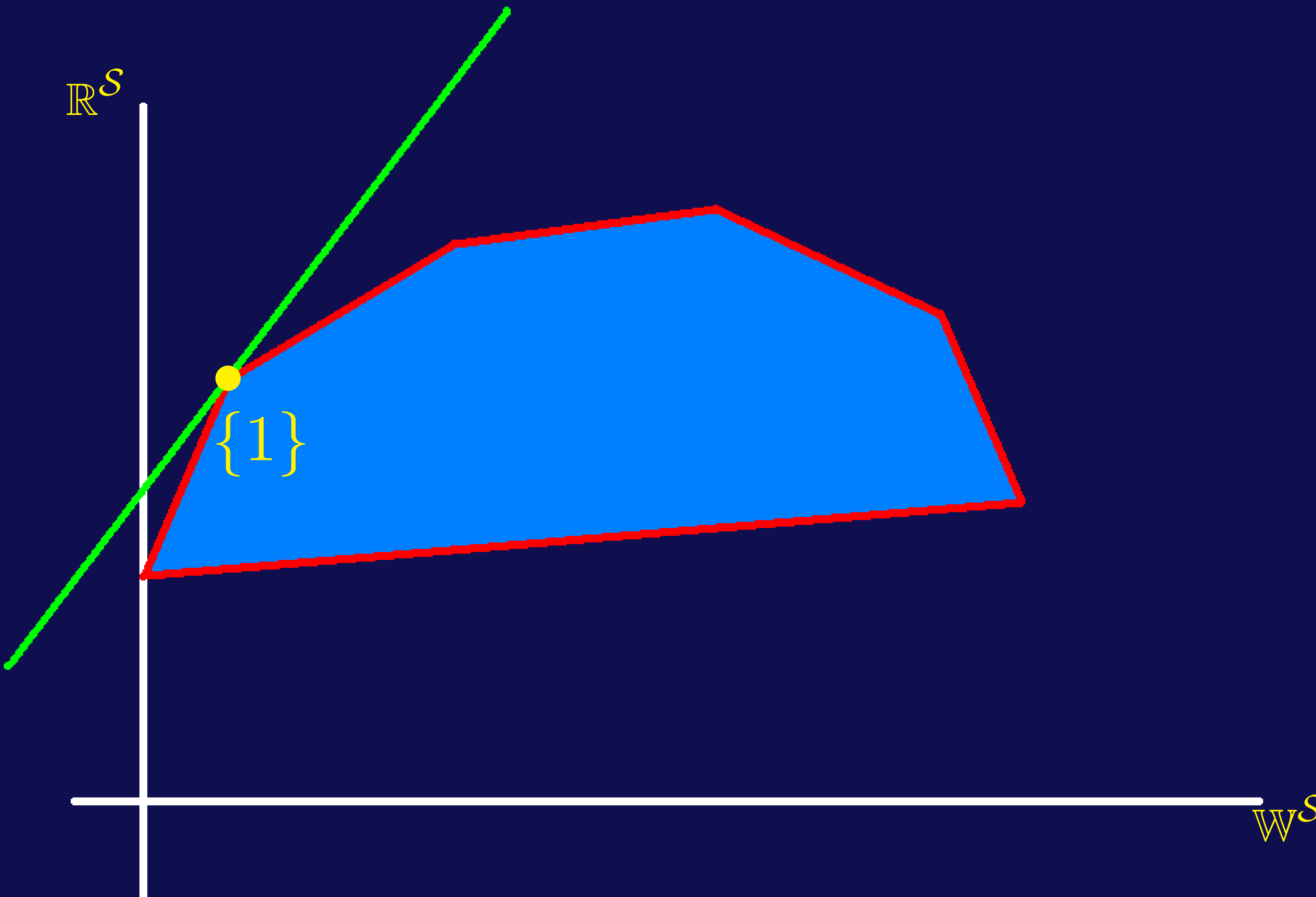
Geometric Interpretation

- $(\mathbb{W}_n^{\mathcal{S}}, \mathbb{R}_n^{\mathcal{S}})$ gives rise to 2-dim. performance region
- Indexability means the performance region is convex
- Optimal (threshold) policies are extreme points of the upper boundary of the performance region
- Index values are slopes of the upper boundary
- Indexability is sort of a dual concept to threshold policies
  - ▷ but not equivalent!

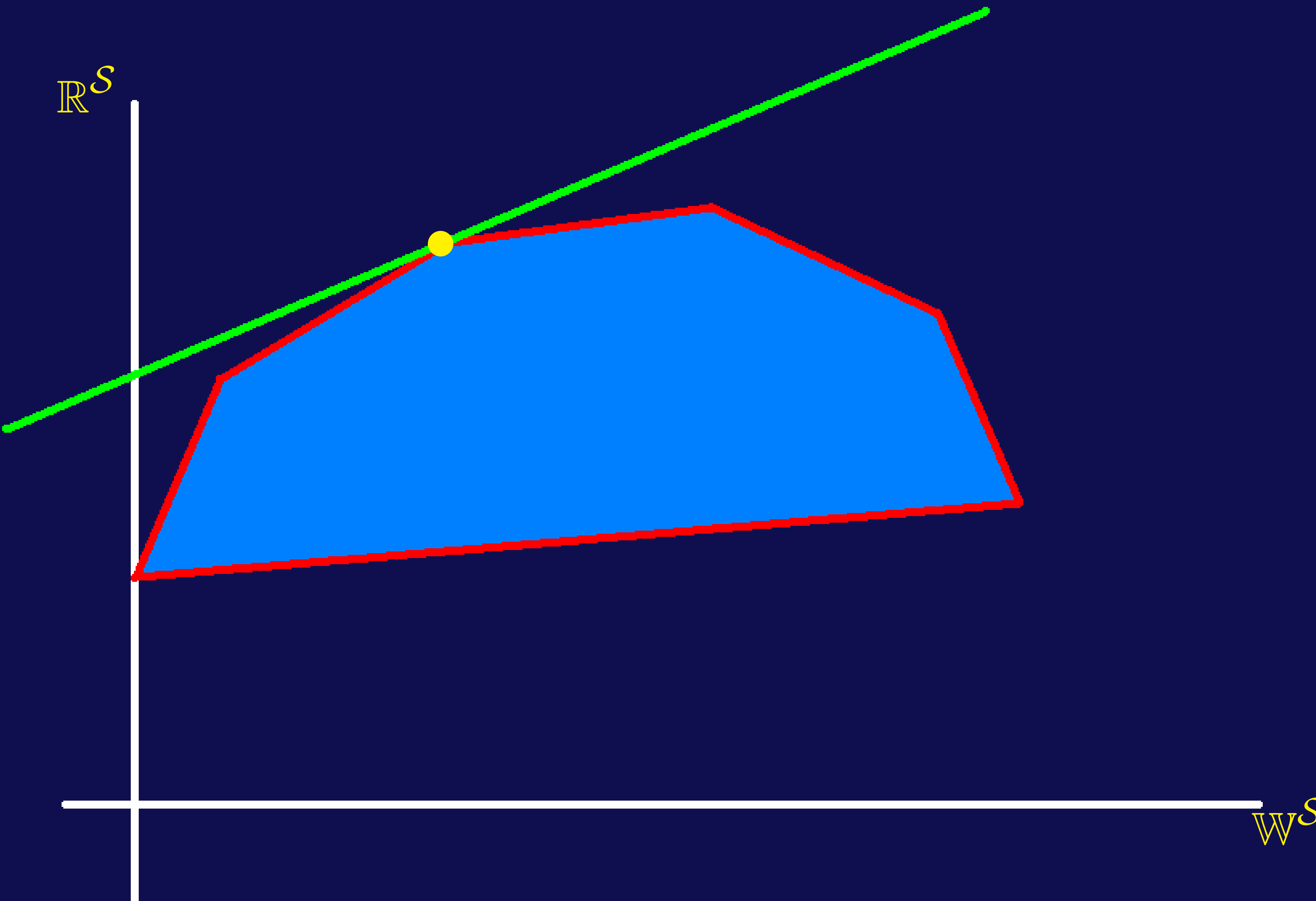
Performance Region



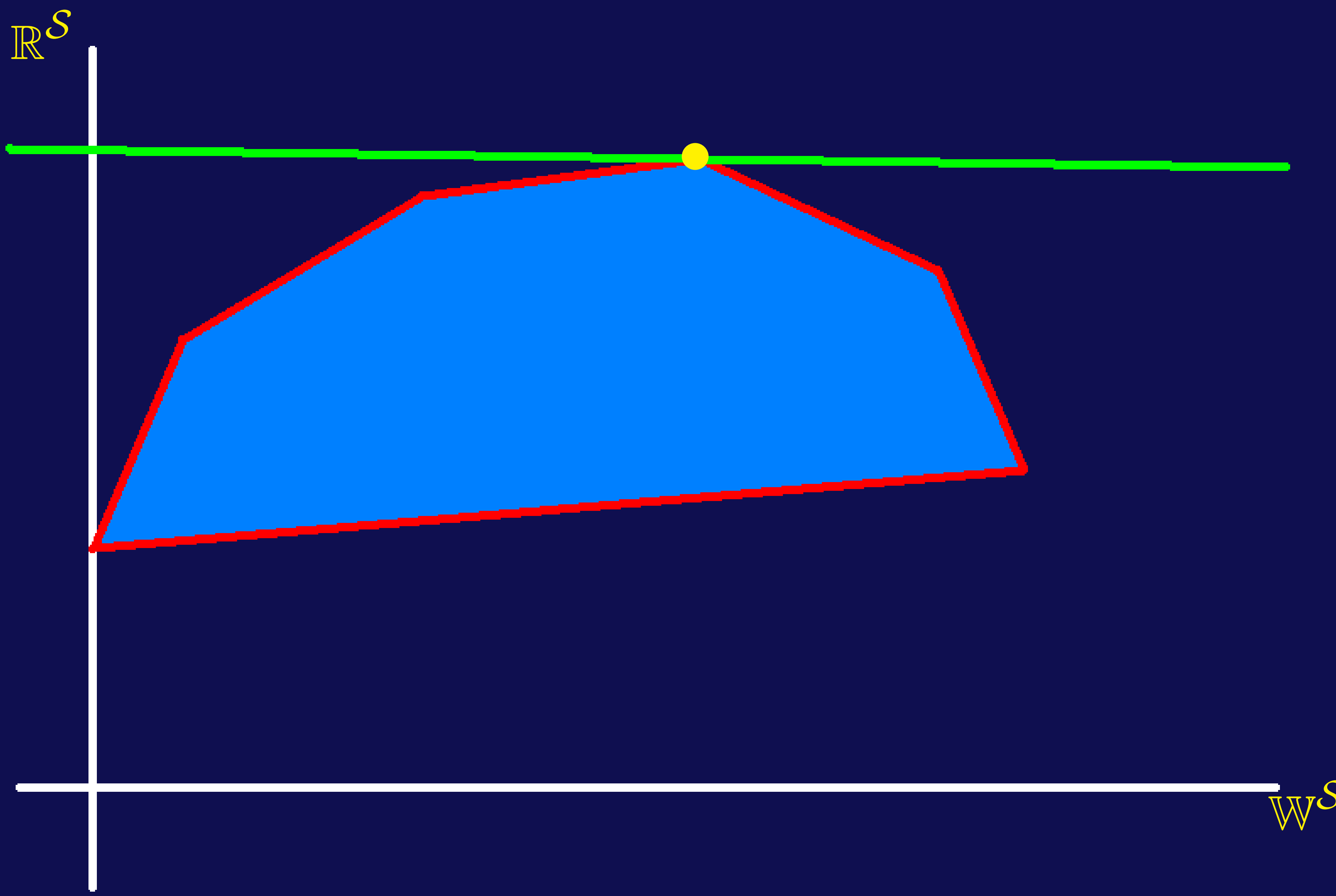
Performance Region



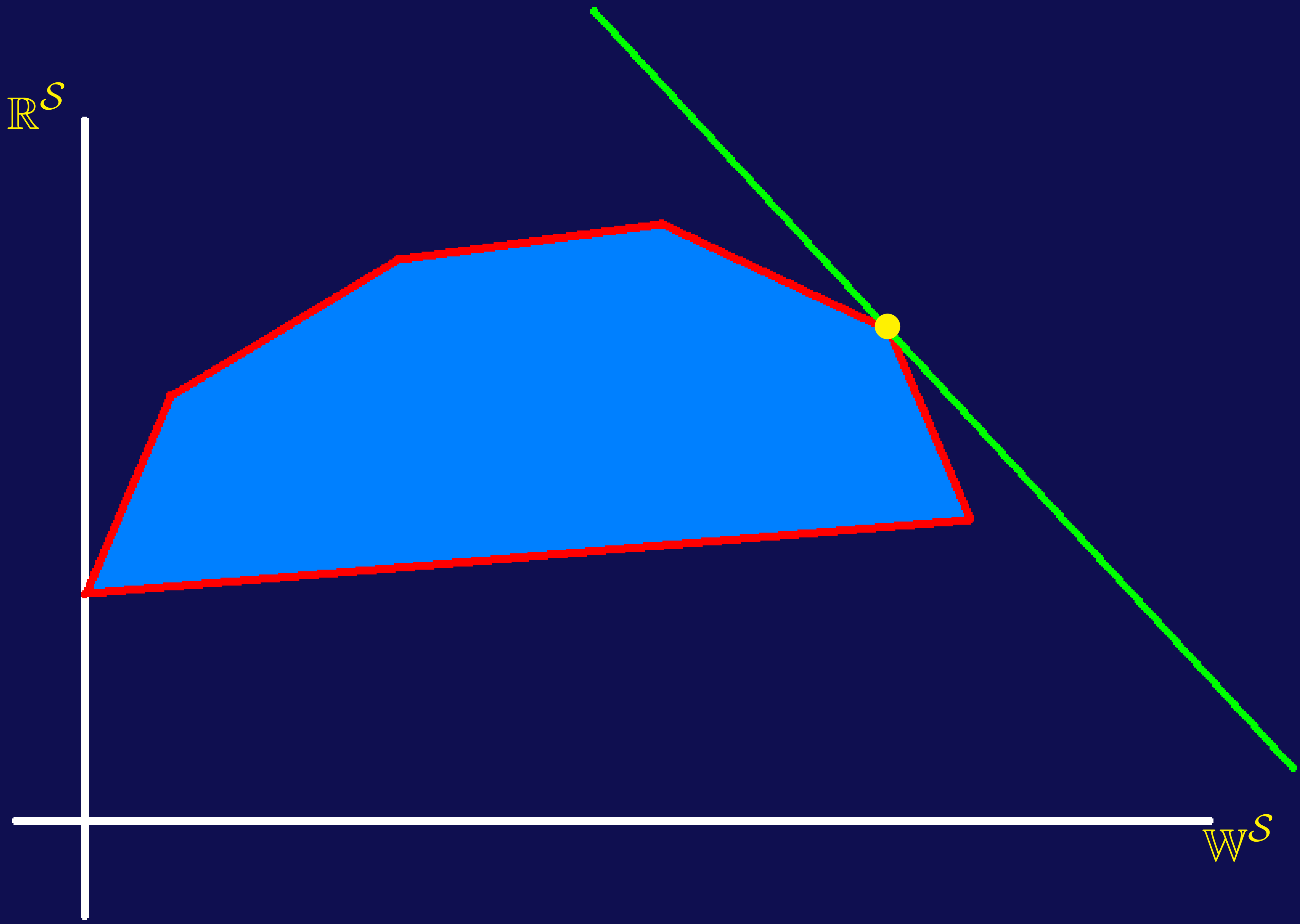
Performance Region



Performance Region



Performance Region



Performance Region