Optimal Scheduling in High Speed Downlink Packet Access

Hussein Zubaidy

November 12, 2006

Objective and Motivation

Methodology

Problem Definition and Model Description

Case Study and Results

Conclusion and Future Work

Objective

To devise a methodology to find the optimal scheduling regime in HSDPA networks, that controls the allocation of the time-code resources.

This policy should have the following properties:

Objective

To devise a methodology to find the optimal scheduling regime in HSDPA networks, that controls the allocation of the time-code resources.

This policy should have the following properties:

- Fair; Divide the resources fairly between all the active users.

Objective

To devise a methodology to find the optimal scheduling regime in HSDPA networks, that controls the allocation of the time-code resources.

This policy should have the following properties:

- Fair; Divide the resources fairly between all the active users.
- Maximizes the overall cell throughput.

Objective

To devise a methodology to find the optimal scheduling regime in HSDPA networks, that controls the allocation of the time-code resources.

This policy should have the following properties:

- Fair; Divide the resources fairly between all the active users.
- Maximizes the overall cell throughput.
- Provide channel aware (diversity gain) and high speed resource allocation.

Motivation

- 3GPP only suggested some guidelines for HSDPA downlink scheduler and left the design specifics undefined.

Motivation

- 3GPP only suggested some guidelines for HSDPA downlink scheduler and left the design specifics undefined.
- This resulted in many different scheduling techniques and implementations most of which are proprietary.

Motivation

- 3GPP only suggested some guidelines for HSDPA downlink scheduler and left the design specifics undefined.
- This resulted in many different scheduling techniques and implementations most of which are proprietary.
- Most of the available work in scheduler design is based on intuition and creativity of the designers. This approach can be described as a *procedural approach* and works as follows:

Motivation cont.

- The designer usually selects an optimization criterion that represents some important performance measure (in his/her opinion) and builds an algorithm based on that criteria.

Motivation cont.

- The designer usually selects an optimization criterion that represents some important performance measure (in his/her opinion) and builds an algorithm based on that criteria.
- Then tries to establish confidence in it using backward analysis or simulation.

Motivation cont.

- The designer usually selects an optimization criterion that represents some important performance measure (in his/her opinion) and builds an algorithm based on that criteria.
- Then tries to establish confidence in it using backward analysis or simulation.
- This, will result in a suboptimal algorithm at the best, that performs well in some scenarios and poor in the others.

Methodology

This work presents a novel approach for scheduling. A declarative approach is used,

Methodology

This work presents a novel approach for scheduling. A declarative approach is used,

- Develop an analytic model for the HSDPA downlink scheduler.

Methodology

This work presents a novel approach for scheduling. A declarative approach is used,

- Develop an analytic model for the HSDPA downlink scheduler.
  - A MDP based discrete stochastic dynamic programming model is used to model the system.

Methodology

This work presents a novel approach for scheduling. A declarative approach is used,

- Develop an analytic model for the HSDPA downlink scheduler.
  - A MDP based discrete stochastic dynamic programming model is used to model the system.
  - This *Model* is a simplifying abstraction of the real scheduler which estimates system behavior under different conditions and describes the role of various system components in these behaviors.

Methodology

This work presents a novel approach for scheduling. A declarative approach is used,

- Develop an analytic model for the HSDPA downlink scheduler.
  - A MDP based discrete stochastic dynamic programming model is used to model the system.
  - This *Model* is a simplifying abstraction of the real scheduler which estimates system behavior under different conditions and describes the role of various system components in these behaviors.
  - Must be solvable.

Methodology (cont.)

- Define an *objective function*.

Methodology (cont.)

- Define an *objective function*.
- Value iteration is then used to solve for optimal policy.

Methodology (cont.)

- Define an *objective function*.
- Value iteration is then used to solve for optimal policy.
- Study the structure of the optimal policy and develop a near-optimal heuristic policy that:

Methodology (cont.)

- Define an *objective function*.
- Value iteration is then used to solve for optimal policy.
- Study the structure of the optimal policy and develop a near-optimal heuristic policy that:
  - Performs close to the Optimal policy.

Methodology (cont.)

- Define an *objective function*.
- Value iteration is then used to solve for optimal policy.
- Study the structure of the optimal policy and develop a near-optimal heuristic policy that:
  - Performs close to the Optimal policy.
  - Has much less computation complexity compared to the value iteration used to determine the optimal policy

Methodology (cont.)

- Define an *objective function*.
- Value iteration is then used to solve for optimal policy.
- Study the structure of the optimal policy and develop a near-optimal heuristic policy that:
  - Performs close to the Optimal policy.
  - Has much less computation complexity compared to the value iteration used to determine the optimal policy
  - Can easily be extended to larger queue sizes.

Problem Definition and Conceptualization

The HSDPA downlink channel uses a mix of TDMA and CDMA:

Problem Definition and Conceptualization

The HSDPA downlink channel uses a mix of TDMA and CDMA:

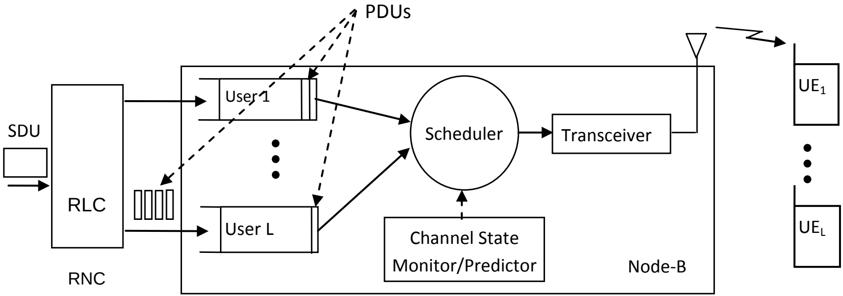
- Time is slotted into fixed length 2 ms TTIs.

Problem Definition and Conceptualization

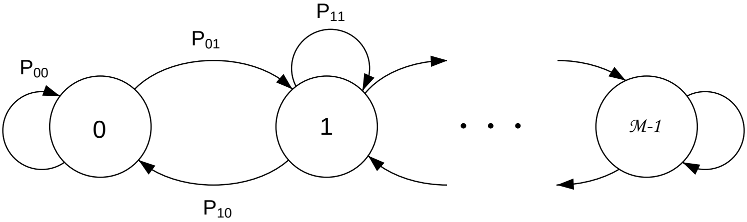
The HSDPA downlink channel uses a mix of TDMA and CDMA:

- Time is slotted into fixed length 2 ms TTIs.
- During each TTI, there are 15 available codes that may be allocated to one or more users.

HSDPA Scheduler Model (Downlink)



FSMC Model for HSDPA Downlink Channel



The Model

- MDP based Model.

The Model

- MDP based Model.
- HSDPA downlink scheduler is modelled by the 5-tuple $(T, S, A, P_{\mathbf{ss'}}(\mathbf{a}), R(\mathbf{s}, \mathbf{a}))$, where,

The Model

- MDP based Model.
- HSDPA downlink scheduler is modelled by the 5-tuple $(T, S, A, P_{\mathbf{ss}'}(\mathbf{a}), R(\mathbf{s}, \mathbf{a}))$, where,
  - T is the set of decision epochs,

The Model

- MDP based Model.
- HSDPA downlink scheduler is modelled by the 5-tuple $(T, S, A, P_{\mathbf{ss'}}(\mathbf{a}), R(\mathbf{s}, \mathbf{a}))$, where,
  - T is the set of decision epochs,
  - S and A are the state and action spaces,

The Model

- MDP based Model.
- HSDPA downlink scheduler is modelled by the 5-tuple $(T, S, A, P_{\mathbf{ss'}}(\mathbf{a}), R(\mathbf{s}, \mathbf{a}))$, where,
  - T is the set of decision epochs,
  - S and A are the state and action spaces,
  - $P_{\mathbf{ss'}}(\mathbf{a}) = Pr(\mathbf{s}(t+1) = \mathbf{s'} | \mathbf{s}(t) = \mathbf{s}, \mathbf{a}(\mathbf{s}) = \mathbf{a})$ is the state transition probability, and

The Model

- MDP based Model.
- HSDPA downlink scheduler is modelled by the 5-tuple $(T, S, A, P_{\mathbf{ss'}}(\mathbf{a}), R(\mathbf{s}, \mathbf{a}))$,
  where,
  - T is the set of decision epochs,
  - S and A are the state and action spaces,
  - $P_{\mathbf{ss'}}(\mathbf{a}) = Pr(\mathbf{s}(t+1) = \mathbf{s'} | \mathbf{s}(t) = \mathbf{s}, \mathbf{a}(\mathbf{s}) = \mathbf{a})$ is the state transition probability, and
  - $R(\mathbf{s}, \mathbf{a})$ is the immediate reward when at state **s** and taking action **a**.

Basic Assumptions

- L active users in the cell.

Basic Assumptions

- L active users in the cell.
- Finite buffer with size B per user for each of the L users.

Basic Assumptions

- L active users in the cell.
- Finite buffer with size B per user for each of the L users.
- Error free transmission.

Basic Assumptions

- L active users in the cell.
- Finite buffer with size B per user for each of the L users.
- Error free transmission.
- SDUs are segmented by RLC into a fixed number of PDUs (u_i) and delivered to Node-B at the beginning of the next TTI.

Basic Assumptions

- L active users in the cell.
- Finite buffer with size B per user for each of the L users.
- Error free transmission.
- SDUs are segmented by RLC into a fixed number of PDUs (u_i) and delivered to Node-B at the beginning of the next TTI.
- For each user $i \in I = \{1, 2, \ldots, L\}$ and slot $t \in T$, we define:

Basic Assumptions

- L active users in the cell.
- Finite buffer with size B per user for each of the L users.
- Error free transmission.
- SDUs are segmented by RLC into a fixed number of PDUs (u_i) and delivered to Node-B at the beginning of the next TTI.
- For each user $i \in I = \{1, 2, \ldots, L\}$ and slot $t \in T$, we define:
  - $y_i(t)$ the number of scheduled PDUs,

Basic Assumptions

- L active users in the cell.
- Finite buffer with size B per user for each of the L users.
- Error free transmission.
- SDUs are segmented by RLC into a fixed number of PDUs (u_i) and delivered to Node-B at the beginning of the next TTI.
- For each user $i \in I = \{1, 2, \ldots, L\}$ and slot $t \in T$, we define:
  - $y_i(t)$ the number of scheduled PDUs,
  - $x_i(t) \in \mathcal{X} = \{0, 1, 2, \ldots, B\}$ the queue size,

Basic Assumptions

- L active users in the cell.
- Finite buffer with size B per user for each of the L users.
- Error free transmission.
- SDUs are segmented by RLC into a fixed number of PDUs (u_i) and delivered to Node-B at the beginning of the next TTI.
- For each user $i \in I = \{1, 2, \ldots, L\}$ and slot $t \in T$, we define:
  - $y_i(t)$ the number of scheduled PDUs,
  - $x_i(t) \in \mathcal{X} = \{0, 1, 2, \ldots, B\}$ the queue size,
  - $z_i(t) \in \{0, u_i\}$ the number of arriving PDUs.

Basic Assumptions

- L active users in the cell.
- Finite buffer with size B per user for each of the L users.
- Error free transmission.
- SDUs are segmented by RLC into a fixed number of PDUs (u_i) and delivered to Node-B at the beginning of the next TTI.
- For each user $i \in I = \{1, 2, \ldots, L\}$ and slot $t \in T$, we define:
  - $y_i(t)$ the number of scheduled PDUs,
  - $x_i(t) \in \mathcal{X} = \{0, 1, 2, \ldots, B\}$ the queue size,
  - $z_i(t) \in \{0, u_i\}$ the number of arriving PDUs.
- Independent Bernoulli arrivals with parameter q_i.

Basic Assumptions

- L active users in the cell.
- Finite buffer with size B per user for each of the L users.
- Error free transmission.
- SDUs are segmented by RLC into a fixed number of PDUs (u_i) and delivered to Node-B at the beginning of the next TTI.
- For each user $i \in I = \{1, 2, \ldots, L\}$ and slot $t \in T$, we define:
  - $y_i(t)$ the number of scheduled PDUs,
  - $x_i(t) \in \mathcal{X} = \{0, 1, 2, \ldots, B\}$ the queue size,
  - $z_i(t) \in \{0, u_i\}$ the number of arriving PDUs.
- Independent Bernoulli arrivals with parameter q_i.
- Scheduler can assign c codes chunks at a time, where $c \in \{1, 3, 5, 15\}$.

Basic Assumptions–FSMC State Space

- The channel state of user i during slot t is denoted by $\gamma_i(t)$.

Basic Assumptions–FSMC State Space

- The channel state of user i during slot t is denoted by $\gamma_i(t)$.
- Channel state space is the set $\mathcal{M} = \{0, 1, \ldots, M-1\}$.

Basic Assumptions–FSMC State Space

- The channel state of user i during slot t is denoted by $\gamma_i(t)$.
- Channel state space is the set $\mathcal{M} = \{0, 1, \dots, M-1\}$.
- user i channel can handle up to $\gamma_i(t)$ PDUs per code.

Basic Assumptions–FSMC State Space

- The channel state of user i during slot t is denoted by $\gamma_i(t)$.
- Channel state space is the set $\mathcal{M} = \{0, 1, \ldots, M-1\}$.
- user i channel can handle up to $\gamma_i(t)$ PDUs per code.
- The Markov transition probability $P_{\gamma_i \gamma_i'}$ is known and can be calculated for any mobile environment with Rayleigh fading channel [Wang and Moayeri].

State

- The system state $\mathbf{s}(t) \in S$ is a vector comprised of multiple state variables

$$\mathbf{s}(t) = (x_1(t), x_2(t), \ldots, x_L(t), \gamma_1(t), \gamma_2(t), \ldots, \gamma_L(t)) \quad (1)$$

State

- The system state $\mathbf{s}(t) \in S$ is a vector comprised of multiple state variables

$$\mathbf{s}(t) = (x_1(t), x_2(t), \dots, x_L(t), \gamma_1(t), \gamma_2(t), \dots, \gamma_L(t)) \quad (1)$$

- $S = \{\mathcal{X} \times \mathcal{M}\}^L$ is finite, due to the assumption of finite buffers size and channel states.

Action Sets

- The action $\mathbf{a}(\mathbf{s}) \in A$ is taken when in state $\mathbf{s}$

$$\mathbf{a}(\mathbf{s}) = (a_1(\mathbf{s}), a_2(\mathbf{s}), \ldots, a_L(\mathbf{s})) \quad (2)$$

Action Sets

- The action $\mathbf{a}(\mathbf{s}) \in A$ is taken when in state $\mathbf{s}$

$$\mathbf{a}(\mathbf{s}) = (a_1(\mathbf{s}), a_2(\mathbf{s}), \ldots, a_L(\mathbf{s})) \tag{2}$$

  - subject to,

$$\sum_{i=1}^{L} a_i(\mathbf{s}) \leq \frac{15}{c}, \quad and \quad a_i(\mathbf{s}) \leq \left\lceil \frac{x_i(t)}{\gamma_i(t)c} \right\rceil$$

Action Sets

- The action $\mathbf{a}(\mathbf{s}) \in A$ is taken when in state $\mathbf{s}$

$$\mathbf{a}(\mathbf{s}) = (a_1(\mathbf{s}), a_2(\mathbf{s}), \ldots, a_L(\mathbf{s})) \tag{2}$$

  - subject to,

$$\sum_{i=1}^{L} a_i(\mathbf{s}) \leq \frac{15}{c}, \quad and \quad a_i(\mathbf{s}) \leq \left\lceil \frac{x_i(t)}{\gamma_i(t)c} \right\rceil$$

  - $a_i(t)c$, number of codes allocated to user i at time epoch t.

Action Sets

- The action $\mathbf{a}(\mathbf{s}) \in A$ is taken when in state $\mathbf{s}$

$$\mathbf{a}(\mathbf{s}) = (a_1(\mathbf{s}), a_2(\mathbf{s}), \ldots, a_L(\mathbf{s})) \tag{2}$$

  - subject to,

$$\sum_{i=1}^{L} a_i(\mathbf{s}) \leq \frac{15}{c}, \quad and \quad a_i(\mathbf{s}) \leq \left\lceil \frac{x_i(t)}{\gamma_i(t)c} \right\rceil$$

  - $a_i(t)c$, number of codes allocated to user i at time epoch t.
- The first constraint means that the policy can not allocate more than the available 15 codes at each time slot.

Action Sets

- The action $\mathbf{a}(\mathbf{s}) \in A$ is taken when in state $\mathbf{s}$

$$\mathbf{a}(\mathbf{s}) = (a_1(\mathbf{s}), a_2(\mathbf{s}), \ldots, a_L(\mathbf{s})) \qquad (2)$$

  - subject to,

$$\sum_{i=1}^{L} a_i(\mathbf{s}) \leq \frac{15}{c}, \quad and \quad a_i(\mathbf{s}) \leq \left\lceil \frac{x_i(t)}{\gamma_i(t)c} \right\rceil$$

  - $a_i(t)c$, number of codes allocated to user i at time epoch t.
- The first constraint means that the policy can not allocate more than the available 15 codes at each time slot.
- The second makes the policy conservative by allocating no more codes to user i than that required to empty its buffer.

Reward Function

- The reward must achieve the **objective function**

Reward Function

- The reward must achieve the **objective function**
- $R(\mathbf{s}, \mathbf{a})$ have two components corresponding to the two objectives

$$R(\mathbf{s}, \mathbf{a}) = \sum_{i=1}^{L} a_i \gamma_i c - \sigma \sum_{i=1}^{L} (B - \bar{x})\, \mathbf{1}_{\{x_i = B\}} \qquad (3)$$

where we defined the **fairness factor** (σ) to reflect the significance of fairness in the optimal policy.

Reward Function

- The reward must achieve the **objective function**
- $R(\mathbf{s}, \mathbf{a})$ have two components corresponding to the two objectives

$$R(\mathbf{s}, \mathbf{a}) = \sum_{i=1}^{L} a_i \gamma_i c - \sigma \sum_{i=1}^{L} (B - \bar{x}) \, \mathbf{1}_{\{x_i = B\}} \tag{3}$$

where we defined the **fairness factor** (σ) to reflect the significance of fairness in the optimal policy.

- The positive term of the reward maximizes the cell throughput.

Reward Function

- The reward must achieve the **objective function**
- $R(\mathbf{s}, \mathbf{a})$ have two components corresponding to the two objectives

$$R(\mathbf{s}, \mathbf{a}) = \sum_{i=1}^{L} a_i \gamma_i c - \sigma \sum_{i=1}^{L} (B - \bar{x})\, \mathbf{1}_{\{x_i = B\}} \qquad (3)$$

where we defined the **fairness factor** (σ) to reflect the significance of fairness in the optimal policy.

- The positive term of the reward maximizes the cell throughput.
- The second term guarantees some level of fairness and reduces dropping probability.

State Transition Probability

- $P_{\mathbf{ss}'}(\mathbf{a})$ denotes the probability that choosing an action $\mathbf{a}$ at time t when in state $\mathbf{s}$ will lead to state $\mathbf{s}'$ at time $t+1$.

State Transition Probability

- $P_{\mathbf{ss}'}(\mathbf{a})$ denotes the probability that choosing an action **a** at time t when in state **s** will lead to state **s**′ at time $t+1$.
-

$$\begin{aligned} P_{\mathbf{ss}'}(\mathbf{a}) &= Pr(\mathbf{s}(t+1)=\mathbf{s}'|\mathbf{s}(t)=\mathbf{s}, \mathbf{a}(t)=\mathbf{a}) \\ &= Pr(x_1', \ldots, x_L', \gamma_1', \ldots, \gamma_L'|x_1, \ldots, x_L, \\ &\quad \gamma_1, \ldots, \gamma_L, a_1, \ldots, a_L) \end{aligned} \quad (4)$$

State Transition Probability cont.

- The evolution of the queue size (x_i) is given by

State Transition Probability cont.

- The evolution of the queue size (x_i) is given by

$$\begin{aligned} x_i' &= \min\left([x_i - y_i]^+ + z_i', B\right) \\ &= \min\left([x_i - a_i\gamma_i c]^+ + z_i', B\right) \end{aligned} \tag{5}$$

State Transition Probability cont.

- The evolution of the queue size (x_i) is given by

$$\begin{aligned} x_i' &= \min\left([x_i - y_i]^+ + z_i', B\right) \\ &= \min\left([x_i - a_i\gamma_i c]^+ + z_i', B\right) \end{aligned} \tag{5}$$

- Using the independence assumption of the channel state and queue sizes, and after some manipulation, we arrived at the following expression

State Transition Probability cont.

- The evolution of the queue size (x_i) is given by

$$\begin{aligned} x_i' &= \min\left([x_i - y_i]^+ + z_i', B\right) \\ &= \min\left([x_i - a_i \gamma_i c]^+ + z_i', B\right) \end{aligned} \tag{5}$$

- Using the independence assumption of the channel state and queue sizes, and after some manipulation, we arrived at the following expression

$$P_{\mathbf{ss}'}(\mathbf{a}) = \prod_{i=1}^{L} \left(P_{x_i x_i'}(\gamma_i, a_i)\, P_{\gamma_i \gamma_i'} \right) \tag{6}$$

State Transition Probability cont.

- The evolution of the queue size (x_i) is given by

$$\begin{aligned} x_i' &= \min\left([x_i - y_i]^+ + z_i', B\right) \\ &= \min\left([x_i - a_i\gamma_i c]^+ + z_i', B\right) \end{aligned} \quad (5)$$

- Using the independence assumption of the channel state and queue sizes, and after some manipulation, we arrived at the following expression
-

$$P_{\mathbf{ss}'}(\mathbf{a}) = \prod_{i=1}^{L} \left(P_{x_i x_i'}(\gamma_i, a_i)\, P_{\gamma_i \gamma_i'} \right) \quad (6)$$

- where $P_{\gamma_i \gamma_i'}$ is the Markov transition probability of the FSMC.

State Transition Probability cont.

We derived $P_{x_i x_i'}(\gamma_i, a_i)$ using (5) and the law of total probability, and arrived at the following expression

State Transition Probability cont.

- $$P_{x_i x_i'}(\gamma_i, a_i) = \begin{cases} 1 & \text{if } x_i' = x_i = B \text{ \& } a_i\gamma_i = 0, \\ q_i & \text{if } x_i' = x_i = B \text{ \& } 0 < a_i\gamma_i c \le u_i, \\ q_i & \text{if } x_i' = B \text{ \& } x_i < B \text{ \& } W1 \ge B, \\ q_i & \text{if } x_i' < B \text{ \& } x_i' = W1, \\ 1 - q_i & \text{if } x_i' < B \text{ \& } x_i' = W2, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

State Transition Probability cont.

- $$P_{x_i x_i'}(\gamma_i, a_i) = \begin{cases} 1 & \text{if } x_i' = x_i = B \ \& \ a_i\gamma_i = 0, \\ q_i & \text{if } x_i' = x_i = B \ \& \ 0 < a_i\gamma_i c \leq u_i, \\ q_i & \text{if } x_i' = B \ \& \ x_i < B \ \& \ W1 \geq B, \\ q_i & \text{if } x_i' < B \ \& \ x_i' = W1, \\ 1 - q_i & \text{if } x_i' < B \ \& \ x_i' = W2, \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$
- where

$$W1 = [x_i - a_i\gamma_i c]^+ + u_i$$
$$W2 = [x_i - a_i\gamma_i c]^+$$

Value Function

- Infinite-horizon MDP.

Value Function

- Infinite-horizon MDP.
- Total expected discounted reward optimality criterion with discount factor λ is used, where $0 < \lambda < 1$.

Value Function

- Infinite-horizon MDP.
- Total expected discounted reward optimality criterion with discount factor λ is used, where $0 < \lambda < 1$.
- The objective is to find the policy π among all policies, that maximize the *value function* $V^{\pi}(s)$.

Value Function

- Infinite-horizon MDP.
- Total expected discounted reward optimality criterion with discount factor λ is used, where $0 < \lambda < 1$.
- The objective is to find the policy π among all policies, that maximize the *value function* $V^{\pi}(s)$.
- The optimal policy is characterized by

$$V^*(\mathbf{s}) = \max_{\mathbf{a} \in A} [R(\mathbf{s}, \mathbf{a}) + \lambda \sum_{\mathbf{s}' \in S} P_{\mathbf{s}\mathbf{s}'}(\mathbf{a}) V^*(\mathbf{s}')] \qquad (8)$$

Value Function

- Infinite-horizon MDP.
- Total expected discounted reward optimality criterion with discount factor λ is used, where $0 < \lambda < 1$.
- The objective is to find the policy π among all policies, that maximize the *value function* $V^{\pi}(s)$.
- The optimal policy is characterized by

$$V^*(\mathbf{s}) = \max_{\mathbf{a} \in A} [R(\mathbf{s}, \mathbf{a}) + \lambda \sum_{\mathbf{s}' \in S} P_{\mathbf{ss}'}(\mathbf{a}) V^*(\mathbf{s}')] \qquad (8)$$

- where, $V^*(\mathbf{s}) = \sup_{\pi} V^{\pi}(\mathbf{s})$, attained when applying the optimal policy π^*.

Value Iteration

- The model was solved numerically using **Value Iteration**. It works as follows

Value Iteration

- The model was solved numerically using **Value Iteration**. It works as follows
  - Define $V_0(\mathbf{s})$ to be any arbitrary bounded function.

Value Iteration

- The model was solved numerically using **Value Iteration**. It works as follows
  - Define $V_0(\mathbf{s})$ to be any arbitrary bounded function.
  - Run the following recursive equation for $n > 0$

$$V_n(\mathbf{s}) = \max_{\mathbf{a} \in A}[R(\mathbf{s}, \mathbf{a}) + \lambda \sum_{\mathbf{s}' \in S} P_{\mathbf{ss}'}(\mathbf{a}) V_{n-1}(\mathbf{s}')]$$

Value Iteration

- The model was solved numerically using **Value Iteration**. It works as follows
  - Define $V_0(\mathbf{s})$ to be any arbitrary bounded function.
  - Run the following recursive equation for $n > 0$

$$V_n(\mathbf{s}) = \max_{\mathbf{a} \in A}[R(\mathbf{s}, \mathbf{a}) + \lambda \sum_{\mathbf{s}' \in S} P_{\mathbf{ss}'}(\mathbf{a}) V_{n-1}(\mathbf{s}')]$$

  - V_n converges to V^* as $n \rightarrow \infty$ [ross].

Value Iteration

- The model was solved numerically using **Value Iteration**. It works as follows
  - Define $V_0(\mathbf{s})$ to be any arbitrary bounded function.
  - Run the following recursive equation for $n > 0$

$$V_n(\mathbf{s}) = \max_{\mathbf{a} \in A}[R(\mathbf{s}, \mathbf{a}) + \lambda \sum_{\mathbf{s}' \in S} P_{\mathbf{ss}'}(\mathbf{a}) V_{n-1}(\mathbf{s}')]$$

  - V_n converges to V^* as $n \to \infty$ [ross].
  - For a given $\epsilon > 0$, the algorithm can be stopped after n iteration, providing the following

$$\|V_{n+1} - V_n\| < \epsilon(1 - \lambda)/2\lambda \qquad (9)$$

Value Iteration

- The model was solved numerically using **Value Iteration**. It works as follows
  - Define $V_0(\mathbf{s})$ to be any arbitrary bounded function.
  - Run the following recursive equation for $n > 0$

$$V_n(\mathbf{s}) = \max_{\mathbf{a} \in A}[R(\mathbf{s}, \mathbf{a}) + \lambda \sum_{\mathbf{s}' \in S} P_{\mathbf{s}\mathbf{s}'}(\mathbf{a}) V_{n-1}(\mathbf{s}')]$$

  - V_n converges to V^* as $n \rightarrow \infty$ [ross].
  - For a given $\epsilon > 0$, the algorithm can be stopped after n iteration, providing the following

$$\| V_{n+1} - V_n \| < \epsilon(1 - \lambda)/2\lambda \qquad (9)$$

- We can generalize for the infinite horizon average reward using results from [puterman] and [ross].

Case Study: Two Users with 2-State FSMC

- Two users (i.e., $L = 2$) sharing the same cell.

Case Study: Two Users with 2-State FSMC

- Two users (i.e., $L = 2$) sharing the same cell.
- The channel is modelled as a two-state FSMC with transition probability matrix

$$\begin{bmatrix} 1-\alpha_i & \alpha_i \\ \beta_i & 1-\beta_i \end{bmatrix} \tag{10}$$

Case Study: Two Users with 2-State FSMC

- Two users (i.e., $L=2$) sharing the same cell.
- The channel is modelled as a two-state FSMC with transition probability matrix

$$\begin{bmatrix} 1-\alpha_i & \alpha_i \\ \beta_i & 1-\beta_i \end{bmatrix} \tag{10}$$

- User i is said to be **connected** when $\gamma_i=1$ with probability $P(\gamma_i=1)=\alpha_i/(\alpha_i+\beta_i)$.

Case Study: Two Users with 2-State FSMC

- Two users (i.e., $L=2$) sharing the same cell.
- The channel is modelled as a two-state FSMC with transition probability matrix

$$\begin{bmatrix} 1-\alpha_i & \alpha_i \\ \beta_i & 1-\beta_i \end{bmatrix} \tag{10}$$

- User i is said to be **connected** when $\gamma_i = 1$ with probability $P(\gamma_i = 1) = \alpha_i / (\alpha_i + \beta_i)$.
- User i is said to be **not connected** $(\gamma_i = 0)$ with probability $P(\gamma_i = 0) = \beta_i / (\alpha_i + \beta_i)$.

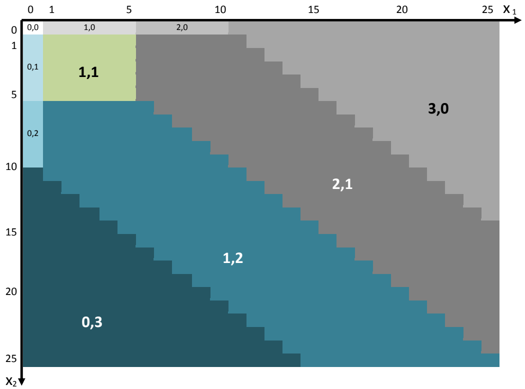
Case Study: Two Users with 2-State FSMC

- Two users (i.e., $L = 2$) sharing the same cell.
- The channel is modelled as a two-state FSMC with transition probability matrix

$$\begin{bmatrix} 1-\alpha_i & \alpha_i \\ \beta_i & 1-\beta_i \end{bmatrix} \tag{10}$$

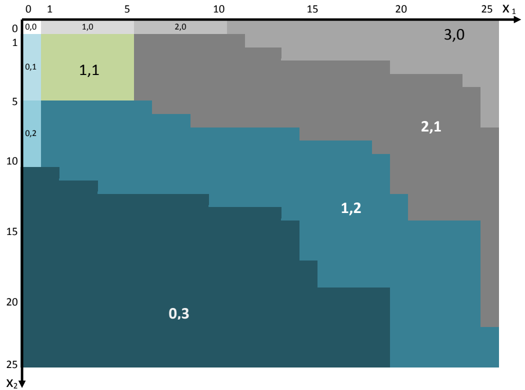
- User i is said to be **connected** when $\gamma_i = 1$ with probability $P(\gamma_i = 1) = \alpha_i/(\alpha_i + \beta_i)$.
- User i is said to be **not connected** ($\gamma_i = 0$) with probability $P(\gamma_i = 0) = \beta_i/(\alpha_i + \beta_i)$.
- $B = 25$, $\sigma = 0.5$, $\lambda = 0.95$, $\epsilon = 0.1$, and $c = 3$, 5 or 15.

The Optimal Policy for Two Symmetrical Users



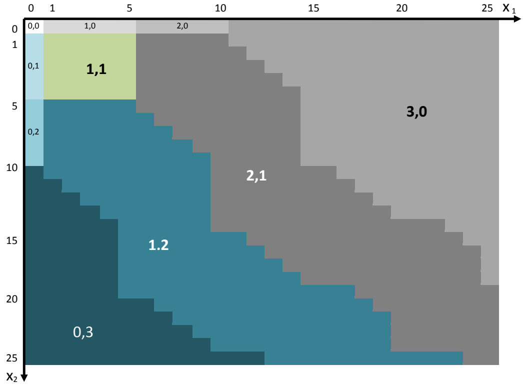
$\alpha_i = \beta_i = p$ for all $0 \leq p \leq 1$ and $P(z_i = 5) = 0.5$ for all $i \in \{1, 2\}$.

The Effect of Channel Quality on Policy Structure



$P(\gamma_1 = 1) = 0.8$ and $P(\gamma_2 = 1) = 0.5$ and $P(z_i = 5) = 0.5$.

The Effect of Arrival Probability on Policy Structure



$$P(\gamma_1 = 1) = P(\gamma_2 = 1) = 0.5 \text{ and } P(z_1 = 5) = 0.8 \; P(z_2 = 5) = 0.5.$$

Heuristic Approach

- Run Value Iteration for small B to find the optimal policy for different scenarios.

Heuristic Approach

- Run Value Iteration for small B to find the optimal policy for different scenarios.
- Study the structure and the behavior of the optimal policy.

Heuristic Approach

- Run Value Iteration for small B to find the optimal policy for different scenarios.
- Study the structure and the behavior of the optimal policy.
- Quantify the effect of channel quality and arrival probability variation on the optimal policy structure.

Heuristic Approach

- Run Value Iteration for small B to find the optimal policy for different scenarios.
- Study the structure and the behavior of the optimal policy.
- Quantify the effect of channel quality and arrival probability variation on the optimal policy structure.
- Use these information to build a heuristic policy that can be extended to larger buffers sizes.

Heuristic Approach

- Run Value Iteration for small B to find the optimal policy for different scenarios.
- Study the structure and the behavior of the optimal policy.
- Quantify the effect of channel quality and arrival probability variation on the optimal policy structure.
- Use these information to build a heuristic policy that can be extended to larger buffers sizes.
- Evaluate the Heuristic policy by comparing the system performance under both policies.

Heuristic Policy

We studied the optimal policy structure by running a wide range of scenarios, we noticed the following trends

Heuristic Policy

We studied the optimal policy structure by running a wide range of scenarios, we noticed the following trends

- The policy is a switch-over and can be described as *share the codes in proportion to the weighted queue length of the connected users*.

Heuristic Policy

We studied the optimal policy structure by running a wide range of scenarios, we noticed the following trends

- The policy is a switch-over and can be described as *share the codes in proportion to the weighted queue length of the connected users*.
- The weight (w_i) is a function of the difference of the two channel qualities and that of the arrival probabilities:

$$w_1 = f([-\Delta P_\gamma]^+, [-\Delta P_z]^+) \quad (11)$$
$$w_2 = f([\Delta P_\gamma]^+, [\Delta P_z]^+) \quad (12)$$

Heuristic Policy cont.

- The intermediate regions has almost a constant width that equals $2c$.

Heuristic Policy cont.

- The intermediate regions has almost a constant width that equals $2c$.
- the optimal policy is monotonic.

Heuristic Policy cont.

- The intermediate regions has almost a constant width that equals $2c$.
- the optimal policy is monotonic.
- a_1 (respectively a_2) is increasing in x_1 (respectively x_2).

Heuristic Policy cont.

- The intermediate regions has almost a constant width that equals $2c$.
- the optimal policy is monotonic.
- a_1 (respectively a_2) is increasing in x_1 (respectively x_2).
- $f()$ is increasing in $|\Delta P_\gamma|$ and decreasing in $|\Delta P_z|$.

Heuristic Policy for $c = 15$

The heuristic policy is a weighted LQF and it assigns codes to users according to the following rules:

Heuristic Policy for $c = 15$

The heuristic policy is a weighted LQF and it assigns codes to users according to the following rules:

- Rule1: when there is only one connected user then assign all the needed codes to that user,

Heuristic Policy for $c = 15$

The heuristic policy is a weighted LQF and it assigns codes to users according to the following rules:

- Rule1: when there is only one connected user then assign all the needed codes to that user,
- Rule2: when both users are not connected (i.e., $\gamma_1 = \gamma_2 = 0$) then no codes will be allocated to any user,

Heuristic Policy for $c = 15$

The heuristic policy is a weighted LQF and it assigns codes to users according to the following rules:

- Rule1: when there is only one connected user then assign all the needed codes to that user,
- Rule2: when both users are not connected (i.e., $\gamma_1 = \gamma_2 = 0$) then no codes will be allocated to any user,
- Rule3: when the two users are connected allocate code chunks according to (13) below

$$\mathbf{a}(t) = \begin{cases} (1,0) & \text{if } w_1 x_1 > w_2 x_2, \\ (0,1) & \text{if } w_1 x_1 \leq w_2 x_2 \end{cases} \qquad (13)$$

Heuristic Policy for $c = 5$

The same heuristic policy above will apply here except for Rule3 which will be modified as follows:

Heuristic Policy for $c = 5$

The same heuristic policy above will apply here except for Rule3 which will be modified as follows:

- Rule3: when the two users are connected, *if* $x_1 + x_2 < 15$ *then* allocate codes to the two users in proportion to their queue length, *else* allocate the code chunks as follows

$$\mathbf{a}(t) = \begin{cases} (3,0) & \text{if } w_1 x_1 > w_2 x_2 + 10, \\ (2,1) & \text{if } w_2 x_2 < w_1 x_1 \leq w_2 x_2 + 10, \\ (1,2) & \text{if } w_2 x_2 - 10 \leq w_1 x_1 \leq w_2 x_2, \\ (0,3) & \text{if } w_1 x_1 < w_2 x_2 - 10, \end{cases} \tag{14}$$

Heuristic Policy for $c = 3$

The heuristics used above can be extended to this case. Again only Rule3 need to be modified as shown below

Heuristic Policy for $c = 3$

The heuristics used above can be extended to this case. Again only Rule3 need to be modified as shown below

- Rule3: when the two users are connected, *if* $x_1 + x_2 < 15$ *then* allocate codes to the two users in proportion to their queue length, *else* allocate the code chunks as follows

$$\mathbf{a}(t) = \begin{cases} (5,0) & \text{if } w_1x_1 > w_2x_2 + 12, \\ (4,1) & \text{if } w_2x_2 + 6 < w_1x_1 \le w_2x_2 + 12, \\ (3,2) & \text{if } w_2x_2 < w_1x_1 \le w_2x_2 + 6, \\ (2,3) & \text{if } w_2x_2 - 6 < w_1x_1 \le w_2x_2, \\ (1,4) & \text{if } w_2x_2 - 12 < w_1x_1 \le w_2x_2 - 6, \\ (0,5) & \text{if } w_1x_1 \le w_2x_2 - 12, \end{cases} \tag{15}$$

Weight Function Approximation

Following these observations, we approximated w_1 and w_2 as follows

$$\hat{w}_1 = 1 + 1.5[-\Delta P_\gamma]^+ - 0.7[-\Delta P_z]^+ \quad (16)$$

$$\hat{w}_2 = 1 + 1.5[\Delta P_\gamma]^+ - 0.7[\Delta P_z]^+ \quad (17)$$

Weight Function Approximation

Following these observations, we approximated w_1 and w_2 as follows

$$\hat{w}_1 = 1 + 1.5[-\Delta P_\gamma]^+ - 0.7[-\Delta P_z]^+ \quad (16)$$
$$\hat{w}_2 = 1 + 1.5[\Delta P_\gamma]^+ - 0.7[\Delta P_z]^+ \quad (17)$$

- where, $\Delta P_\gamma = P(\gamma_1 = 1) - P(\gamma_2 = 1)$

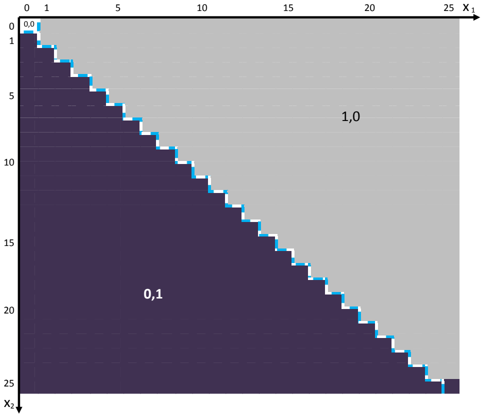
Weight Function Approximation

Following these observations, we approximated w_1 and w_2 as follows

$$\hat{w}_1 = 1 + 1.5[-\Delta P_\gamma]^+ - 0.7[-\Delta P_z]^+ \quad (16)$$
$$\hat{w}_2 = 1 + 1.5[\Delta P_\gamma]^+ - 0.7[\Delta P_z]^+ \quad (17)$$

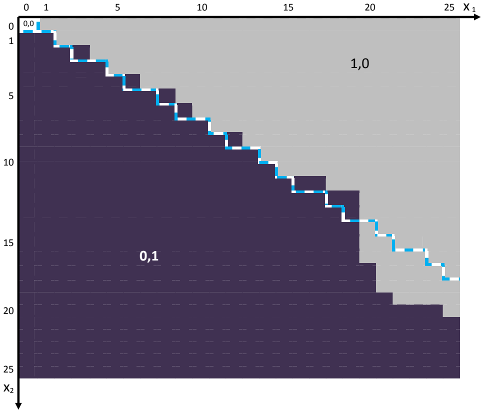
- where, $\Delta P_\gamma = P(\gamma_1 = 1) - P(\gamma_2 = 1)$
- and $\Delta P_z = P(z_1 = u) - P(z_2 = u)$.

Heuristic (dotted line) vs. optimal policy; $c = 15$



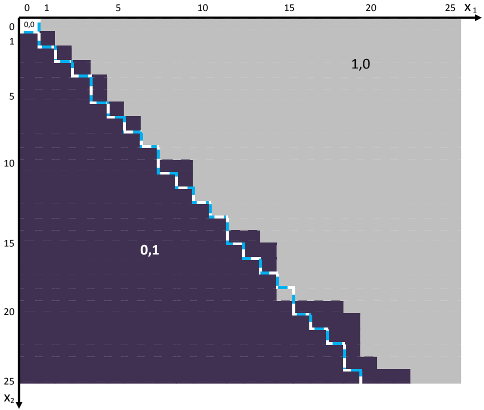
$\alpha_i = \beta_i = p$ for all $0 \leq p \leq 1$ and $P(z_i = 5) = 0.5$ for all $i \in \{1, 2\}$.

Heuristic (dotted line) vs. optimal policy; $c = 15$



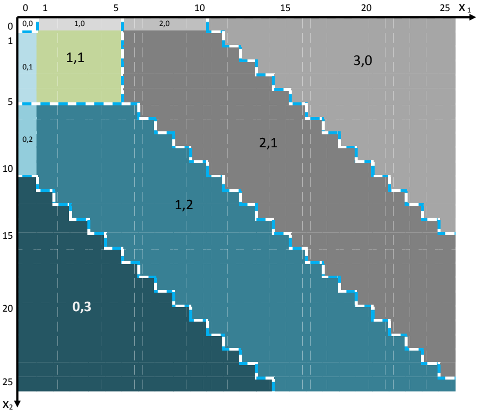
$P(\gamma_1 = 1) = 0.8$ and $P(\gamma_2 = 1) = 0.5$ and $P(z_i = 5) = 0.5$.

Heuristic (dotted line) vs. optimal policy; $c = 15$



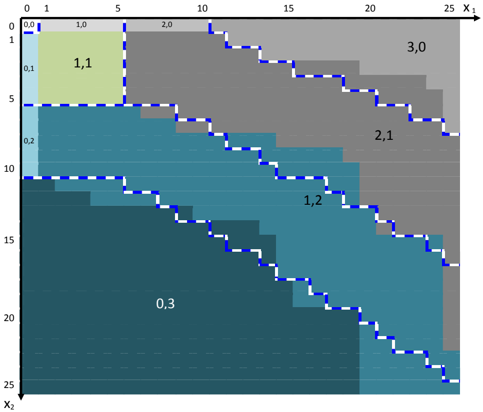
$$P(\gamma_1 = 1) = P(\gamma_2 = 1) = 0.5 \text{ and } P(z_1 = 5) = 0.8\ P(z_2 = 5) = 0.5.$$

Heuristic (dotted line) vs. optimal policy; $c = 5$



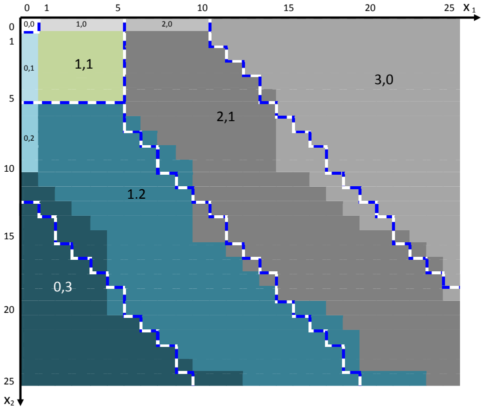
$\alpha_i = \beta_i = p$ for all $0 \leq p \leq 1$ and $P(z_i = 5) = 0.5$ for all $i \in \{1, 2\}$.

Heuristic (dotted line) vs. optimal policy; $c = 5$



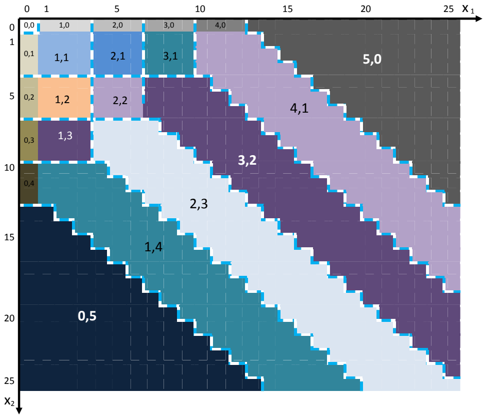
$$P(\gamma_1 = 1) = 0.8 \text{ and } P(\gamma_2 = 1) = 0.5 \text{ and } P(z_i = 5) = 0.5.$$

Heuristic (dotted line) vs. optimal policy; $c = 5$



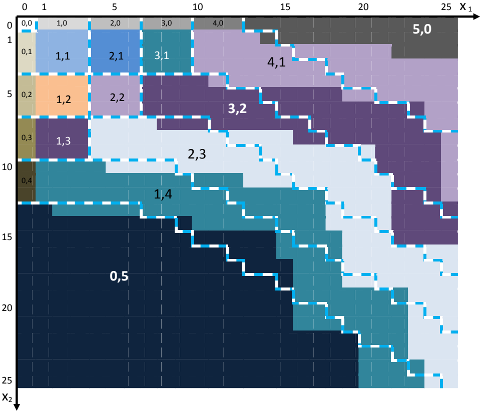
$P(\gamma_1 = 1) = P(\gamma_2 = 1) = 0.5$ and $P(z_1 = 5) = 0.8$ $P(z_2 = 5) = 0.5$.

Heuristic (dotted line) vs. optimal policy; $c = 3$



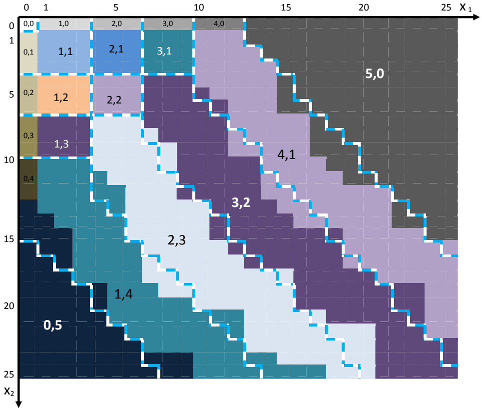
$\alpha_i = \beta_i = p$ for all $0 \leq p \leq 1$ and $P(z_i = 5) = 0.5$ for all $i \in \{1, 2\}$.

Heuristic (dotted line) vs. optimal policy; $c = 3$



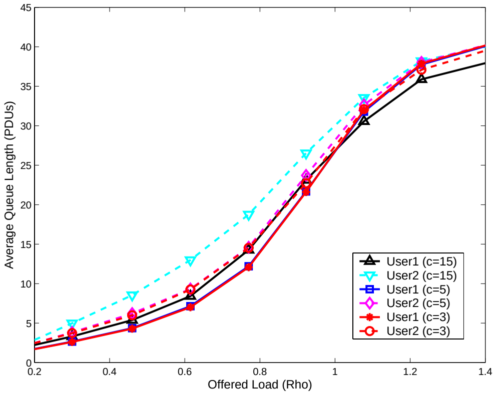
$P(\gamma_1 = 1) = 0.8$ and $P(\gamma_2 = 1) = 0.5$ and $P(z_i = 5) = 0.5$.

Heuristic (dotted line) vs. optimal policy; $c = 3$

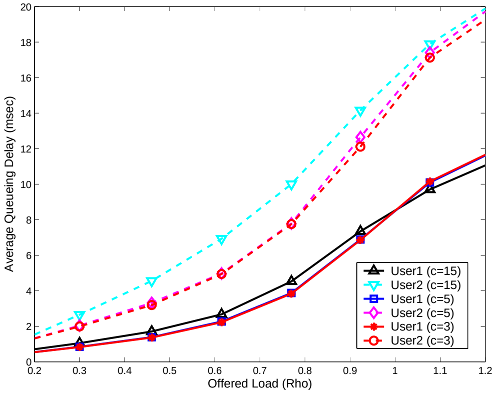


$P(\gamma_1 = 1) = P(\gamma_2 = 1) = 0.5$ and $P(z_1 = 5) = 0.8$ $P(z_2 = 5) = 0.5$.

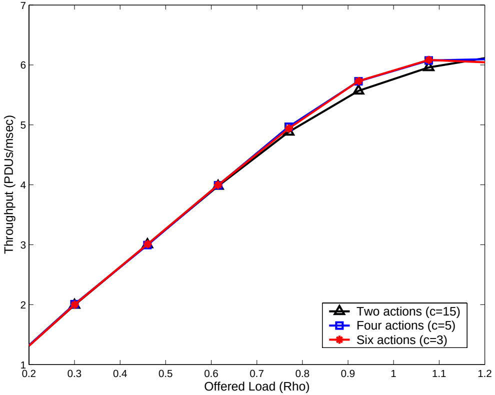
The Effect of Policy Granularity on Queue Length



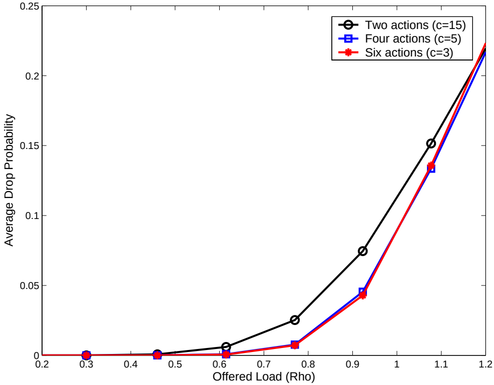
Effect of Policy Granularity on Ave. Queueing Delay



The Effect of Policy Granularity on Scheduler Throughput



The Effect of Policy Granularity on Dropping Probability



Heuristic Policy Evaluation

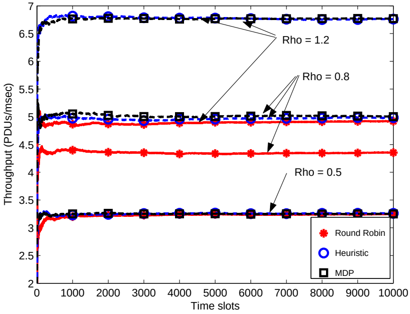


Figure: System throughput for different loading conditions.

Where $\rho = \sum_i P_{z_i} u_i / r^{\pi}$ is the offered load and r^{π} is the measured system capacity under the policy π.

Heuristic Policy Evaluation

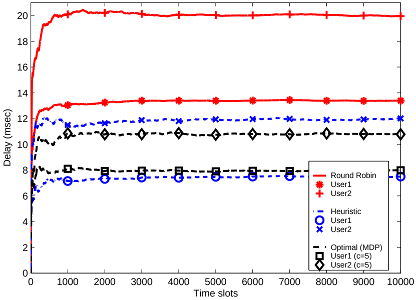


Figure: Queueing delay performance, $P(\gamma_1 = 1) = 0.84$, $P(\gamma_2 = 1) = 0.5$, $q_1 = 0.8$, $q_2 = 0.5$ and $u = 10$.

Heuristic Policy Evaluation

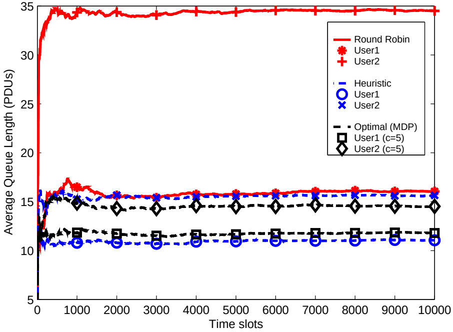


Figure: Queue length, $\rho = 0.75$, $P(\gamma_1 = 1) = 0.84$, $P(\gamma_2 = 1) = 0.5$, $q_1 = 0.5$, $q_2 = 0.5$ and $u = 10$.

Heuristic Policy Evaluation

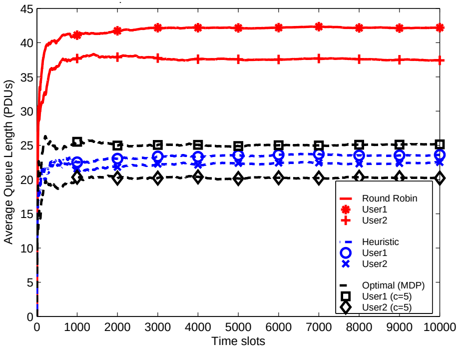


Figure: Queue length, $\rho = 1.0$, $P(\gamma_1 = 1) = 0.84$, $P(\gamma_2 = 1) = 0.5$, $q_1 = 0.8$, $q_2 = 0.5$ and $u = 10$.

Conclusion

- The optimal policy can be described as *share the codes in proportion to the weighted queue length of the connected users*.

Conclusion

- The optimal policy can be described as *share the codes in proportion to the weighted queue length of the connected users*.
- A policy with finer granularity will perform better in light to moderate loading conditions, while a coarse policy is more desirable in heavy loading conditions.

Conclusion

- The optimal policy can be described as *share the codes in proportion to the weighted queue length of the connected users*.
- A policy with finer granularity will perform better in light to moderate loading conditions, while a coarse policy is more desirable in heavy loading conditions.
- However, the performance gain when using $c < 5$ is marginal and does not justify the added complexity.

Conclusion cont.

- The suggested heuristic policy has a reduced constant time complexity ($O(1)$) as compared to the exponential time complexity needed in the determination of the optimal policy.

Conclusion cont.

- The suggested heuristic policy has a reduced constant time complexity ($O(1)$) as compared to the exponential time complexity needed in the determination of the optimal policy.
- The performance of the resulted heuristic policy matches very closely to the optimal policy.

Conclusion cont.

- The suggested heuristic policy has a reduced constant time complexity ($O(1)$) as compared to the exponential time complexity needed in the determination of the optimal policy.
- The performance of the resulted heuristic policy matches very closely to the optimal policy.
- The results also proved that RR is undesirable in HSDPA system due to the poor performance and lack of fairness.

Conclusion cont.

- The suggested heuristic policy has a reduced constant time complexity ($O(1)$) as compared to the exponential time complexity needed in the determination of the optimal policy.
- The performance of the resulted heuristic policy matches very closely to the optimal policy.
- The results also proved that RR is undesirable in HSDPA system due to the poor performance and lack of fairness.
- The suggested heuristic policy can be extended to the case with more than two active users. It also can be easily adapted to accommodate more than one class of service.

Future Work

- Using the proposed model to study a system with more than 2 users.

Future Work

- Using the proposed model to study a system with more than 2 users.
- Using the proposed model to study a system with 3-state FSMC or higher.

Future Work

- Using the proposed model to study a system with more than 2 users.
- Using the proposed model to study a system with 3-state FSMC or higher.
- Relax the assumption of error free transmission and extend the model to take into account retransmissions.

Future Work

- Using the proposed model to study a system with more than 2 users.
- Using the proposed model to study a system with 3-state FSMC or higher.
- Relax the assumption of error free transmission and extend the model to take into account retransmissions.
- Study the effect of using different arrival process statistics using simulation obviously.

Future Work cont.

- Evaluate the Heuristic policy by finding the value function when using the heuristic policy as input and compare it to the value function we obtained from value iteration.

Future Work cont.

- Evaluate the Heuristic policy by finding the value function when using the heuristic policy as input and compare it to the value function we obtained from value iteration.
- Prove analytically some of the optimal policy and value function characteristics, such as monotonicity, multi-modularity, and the switch-over behavior that we noticed before.

Thank You

Discussion

Acronyms

- HSDPA–High Speed Downlink Packet Access.
- 3GPP–Third Generation Partnership Project
- MDP–Markov Decision Process
- TDMA–Time Division Multiple Access
- CDMA–Code Division Multiple Access
- TTI–Transmission Time Interval (2 ms)
- FSMC–Finite State Markov Channel
- SDU–Service Data Unit
- RLC–Radio Link Control Protocol located at Radio Network Controller (RNC)
- PDU–Protocol data unit
- LQF–Longest Queue First