# Tutorial on Intelligent Agents for Cognitive Radios

Haris Volos, Research Assistant Professor
Department of Electrical and Computer Engineering
The University of Arizona

# Outline

1. Cognitive Radio Basics
2. Intelligent Agents for Cognitive Radio
3. Simulation Model
4. Metacognition and Cognitive Engines
5. Summary of Methods Used in Cognitive Engines
6. More Details and Results in Methods Used

# COGNITIVE RADIO BASICS

# Cognitive Radio Definition

- Joe Mitolla “a really smart radio that would be self-aware, RF-aware, user-aware, and that would include language technology and machine vision along with a lot of high-fidelity knowledge of the radio environment.”

*Source: Patric Mannion, “Smart radios stretch spectrum,” EE Times, Dec. 5, 2005*
*http://www.eetimes.com/document.asp?doc_id=1157956*

# Radio Cognition Capability Levels

| Level | Capability | Task Characteristics |
|---|---|---|
| 0 | Pre-programmed | The radio has no model-based reasoning capability |
| 1 | Goal-driven | Goal-driven choice of RF band, air interface, and protocol |
| 2 | Context Awareness | Infers external communications context (minimum user involvement) |
| 3 | Radio Aware | Flexible reasoning about internal and network architectures |
| 4 | Capable of Planning | Reasons over goals as a function of time, space, and context |
| 5 | Conducts Negotiations | Expresses arguments for plans/ alternatives to user, peers, networks |
| 6 | Learns Fluents | Autonomously determines the structure of the environment |
| 7 | Adapts Plans | Autonomously modifies plans as learned fluents change |
| 8 | Adapts Protocols | Autonomously proposes and negotiates new protocols |

*Source: J. Mitola III, "Cognitive Radio: Model-Based Competence for Software Radio," Licentiate Thesis, The Royal Institute of Technology (KTH), Stockholm , Sweden, 1999.*

# Cognitive Radio Applications

- Dynamic spectrum access
- Dynamic spectrum markets
- Spectrum management
- Link adaptation
- Communications in adverse conditions
- Electronic warfare

# Research Opportunities

- Validation and verification for cognitive radios
- Intelligent agent design methods
  - Learning methods
  - Architectures
  - Meta-cognitive methods
- Signal detection and classification
- SDR platforms and architectures for CR
  - Multiband receivers
  - Multiband antennas

# Research Opportunities

- Meta-Cognition
  - A CE for CEs
  - How to manage a group of CEs?
    - Which one is best for each situation?
    - How to characterize them?
- How to compartmentalize a CE's functions and components?
  - How to evaluate a compartmentalized CE?
- Coping with imperfect observations
  - Delayed
  - Corrupted
  - Malicious
- Decision making based on large amount of information
  - What is relevant or irrelevant for each decision?

# Meta-Cognition

- Meta-cognition is defined as "cognition about cognition", or "knowing about knowing". It can take many forms, it includes knowledge about when and how to use particular strategies for learning or for problem solving.

- **What can Meta cognition do in CR?**
  - Provide evaluation methods for CE techniques
  - Improve the CE techniques utilization by monitoring them
  - Using various CE techniques with different abilities at the appropriate times.

Our current work on Meta-Cognition:

Session 5A: CR and DSA Architectures and Systems II at *IC Large Auditorium*

**Learning Characterization Framework and Analysis for a Meta-Cognitive Radio Engine**

# INTELLIGENT AGENTS FOR COGNITIVE RADIO

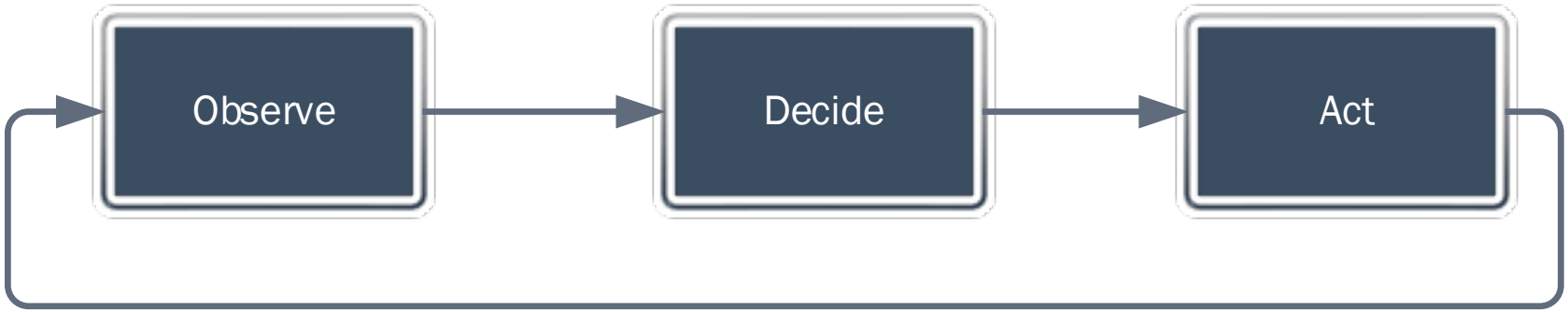
# Intelligent Agents for Cognitive Radio

- **Definition**

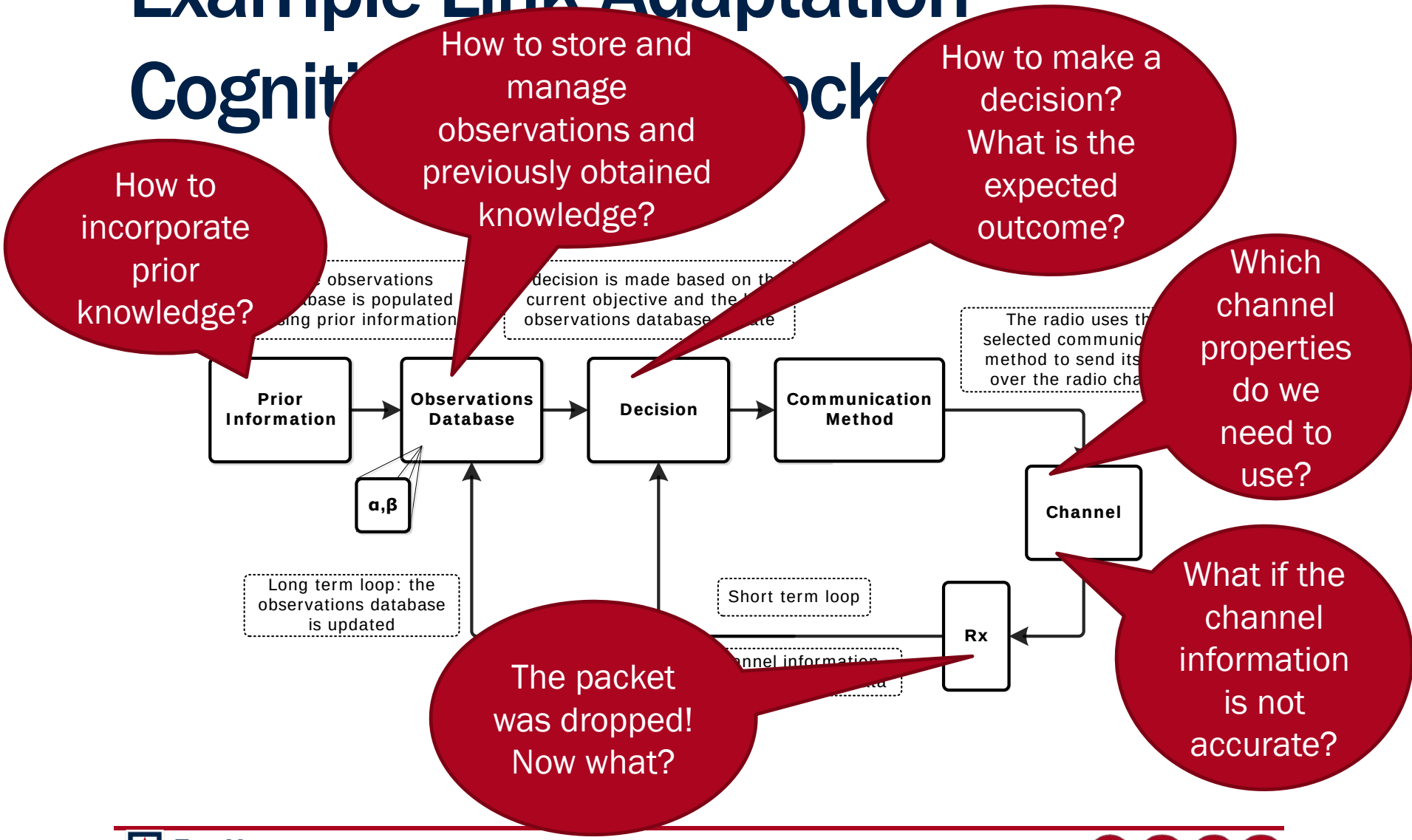
  An IA enables the radio to have the desired learning and adaptation abilities. The IA senses its environment (the wireless channel), acts by using a communication method based on its past experience, and observes its own performance to learn its capabilities, adding to its experience base.

- An IA for CR is commonly known as a “Cognitive Radio Engine,” or just “Cognitive Engine.”

# Basic Cognition Cycle



# Example Link Adaptation Cognit… …ock

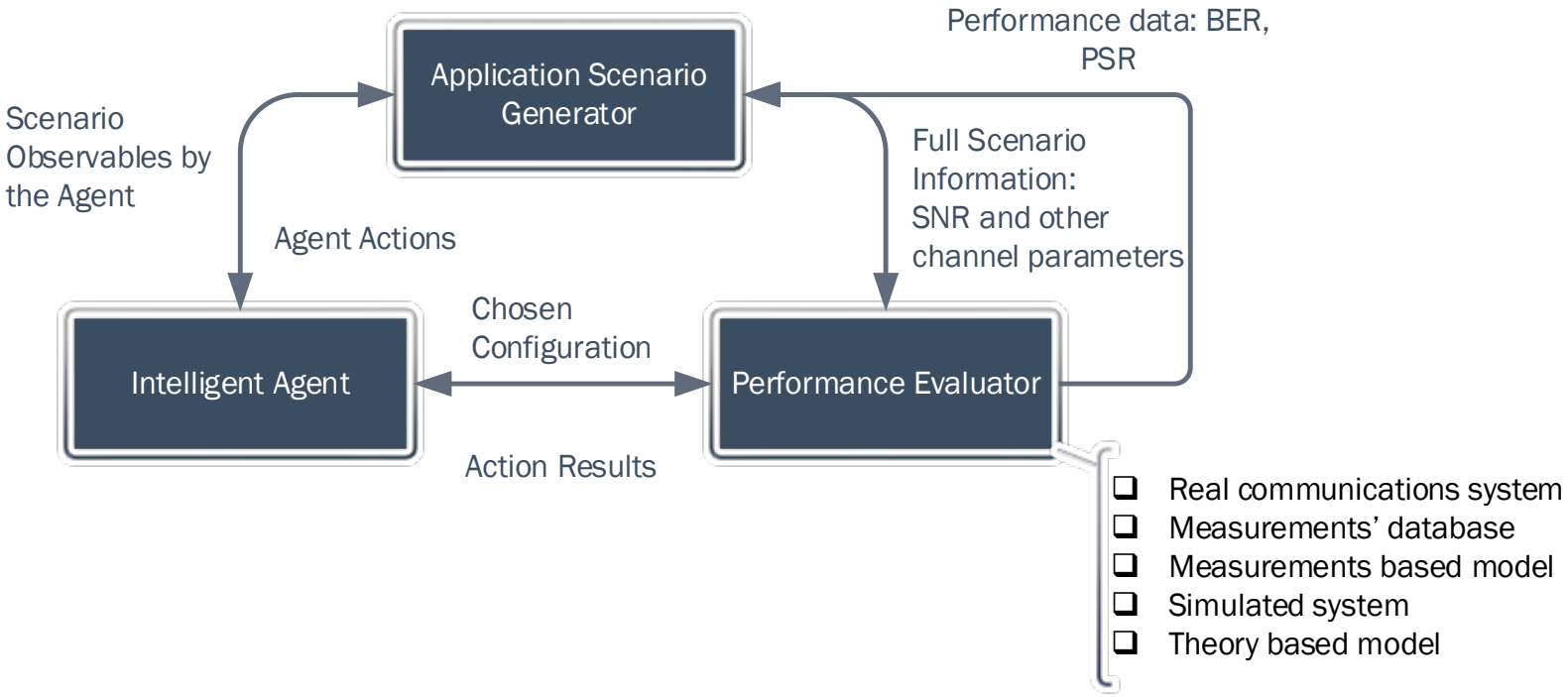


# Cognitive Engine Memory

- Fixed relationships (need to be learned only once)
  - Error vs. SNR
- Optimization rules
  - Solutions for different objectives
  - Solutions that satisfy different policies
  - Solutions for specific emitters
- Volatile
  - Spectrum occupancy parameters
  - Primary user behavior

# SIMULATION MODEL

# Simulating an IA



# Symbol Error Rate Bounds and Approximations

| | |
|---|---|
| M-PSK[b] | $P_s = 2Q\left(\sqrt{2\frac{\gamma_S M_r}{N_t} g_c} sin\left(\frac{\pi}{M}\right)\right)$ |
| M-QAM | $P_s = 1 - \left[1 - 2\left(1 - \frac{1}{\sqrt{M}}\right) Q\left(\sqrt{\frac{3}{M-1}\frac{\gamma_S M_r}{N_t} g_c}\right)\right]^2$ |
| M-QAM[c] | $P_s \leq 1 - \left[1 - 2Q\left(\sqrt{\frac{3}{M-1}\frac{\gamma_S M_r}{N_t} g_c}\right)\right]^2$ |
| MRC[d] | $P_s \leq M_n \prod_{i=1}^{N_t M_r} \frac{1}{1 + \left(\gamma_S g_c \frac{d_{min}^2}{4N_t} \lambda_i'(R)\right)}$ |
| STBC[e] | Same as MRC with $\gamma_S \equiv r\gamma_S$ |
| V-BLAST[f] | $P_s \leq M_n \left(\frac{\gamma_S g_c d_{min}^2 \lambda_{min}}{2M_T}\right)^{-(M_r - N_t + 1)}$ |

[a]adapted from [12] & [15]. [b]$g_c$ is the coding gain. [c]odd $log_2 M$. [d]$M_n$ is the max. # of neighboring symbols within $d_{min}$. $\lambda_i'$ is the $i$th eigen-value of $R = E\left\{vec(H)\, vec(H)^H\right\}$. [e]$r$ is the rate of the STBC code. [f]$\lambda_{min}$ is the smallest eigen-value of $HH^H$.


[12] J. Proakis, *Digital Communications*. New York: McGraw-Hill, 4 ed., 2001.

[15] A. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-Time Wireless Communications*. Cambridge: Cambridge University Press, 2003.

# Confidence Intervals

## Packet Success Rate

It can be shown that

$$p(\theta|\alpha) = Be(\theta; \alpha + \alpha_0 + 1, n - \alpha + \beta_0 + 1)$$

where $\theta$ is the PSR, $n$ the number of total trials, $\alpha$ the number of successful trials, and $\alpha_0$ and $\beta_0$ the prior successes and failures respectively

## Confidence Intervals

$$I_{\theta_l}(\alpha_0 + \alpha, \beta_0 + n - \alpha) = \frac{\delta}{2}$$

and

$$I_{\theta_u}(\alpha_0 + \alpha, \beta_0 + n - \alpha) = 1 - \frac{\delta}{2}$$

where $I_x(\alpha, \beta)$ is the regularized incomplete Beta function.

# SUMMARY OF METHODS USED IN COGNITIVE ENGINES

# Methods used in Cognitive Engines

- Learning Techniques:
  - Epsilon-Greedy Exploration
  - Boltzmann's Exploration
  - Gittins's Index
  - Robust Training Algorithm
  - Bayesian methods (classification; confidence intervals)
  - Neural Networks
  - Predicate logic
- Optimization Techniques:
  - Evolutionary based algorithms (e.g. Genetic algorithm)
  - Other search methods
- Frameworks:
  - Case-Based Reasoning

Balance Exploration vs. Exploitation

Long term optimal performance

Learn while maintaining a min perf. level

Structured way for managing past experience (cases)

# Genetic Algorithm

- Search technique influenced by biological evolution.
- Goal: to find a set of parameters (genes) that maximize an objective (fitness) function
- Steps of a typical GA:
  1. Initial set of solutions (population)
  2. Evaluate population members' fitness
  3. Stochastically, based on their fitness, it selects a number of members to perform a crossover and possibly a mutation operation.
     - A crossover operation combines genes from two members (parents) to generate an offspring.
     - A mutation operation takes a member or an offspring and randomly changes one of its genes.
  4. The fitness of the new members is evaluated.
  5. Based on the fitness level and/or the number of generations, the GA stops and returns the fittest member as the solution.

# Genetic Algorithm Continued

- Several CE's are based on a GA such as [4–11]. Some variations exist that may offer some advantages in certain applications, for example, quantum encoding of the genes expands the search range and the effectiveness of the GA [10]. Likewise, a GA based on chaotic theory [7] exhibits lower similarity among the members and overall improved searching effectiveness.
- ***Pros:*** Straightforward to implement for optimizing multi-objective functions over large search spaces.
- ***Cons:*** Although it can facilitate search over large spaces, is not very well suited to handle stochastic fitness functions and manage the exploration (try new methods) rate versus using proven methods.
- Other related approaches: ant-colony optimization [12] (also evolutionary based) ; animal flocking based such as particle swarm optimization [13].

# Case Based Reasoning

- CBR is a methodology for solving new problems by adapting solutions to older problems.
- A problem can be a set of channel observations and available communication techniques, and the solution can be the communication technique or parameters that achieve the desired goal e.g. maximum channel capacity.
- The CBR methodology assumes that the system has memory of all the previous problems and solutions.
- The past cases of problem-solution pairs can be used as the base of solving of new similar problems.
- A CBR system has the following life cycle:
  1. Retrieve previous cases similar to the problem,
  2. Reuse cases that solve the current problem or adapt similar cases to solve the problem, and
  3. Retain the solution in memory[14].

# Case Based Reasoning Continued

- Examples of CBR include [15], [16] also CBR is often used along with a GA [4], [10], [17].
- ***Pros:*** Intuitive and effective approach that provides a structured way of providing memory to the CE. A CBR is very effective for policy engines [18].
- ***Cons:*** The CBR is just a framework; the designer needs to develop the similarity and adaptation methods that best fit his or her application.
- Care is needed so that the CBR implementation will be able to handle stochastic events (e.g. the successful transmission of a wireless data packet.)

# Artificial Neural Networks

- An ANN is an information processing system inspired by the way human brain neurons work [19].
- An ANN is an interconnected group of artificial neurons.
- Typically, an ANN is used to learn an unknown function or to recognize certain patterns.
- Learning is usually done by training the ANN with pairs of input and output examples.
  - Training is done using an appropriate training algorithm for the architecture of the ANN.
- In CR, ANN are used for learning the adaptation rules [11], [20]; in particular, in [11] a GA is used to provide the examples for the ANN. Other ANN based works are [21–23]
- ***Pros:*** Can learn arbitrary relationships between input and output pairs. Very useful for challenging learning situations that other approaches many not work well.
- ***Cons:*** ANNs require extensive training and examples. They do not provide insights on why certain outputs are chosen.

# Bayesian Methods

- The Bayes rule [3] provides an elegant way to estimate the posterior probability of an event A given that vent B occurred using the of prior probability of A occurring and the conditional probability of A occurring A given that be B occurred.
- The Bayes rule can be used as a classifier [24] or to estimate confidence intervals [25] by combining prior information. Other hypothesis testing approaches for CR can be found at [27].
- ***Pros:*** Solid method for incorporating all the information available from current and past observations.
- ***Cons:***
  - If the application has many variables, the estimation of the conditional probabilities for all the combinations can quickly get very computationally expensive.
  - If some of the variables are independent (or weakly depended) to each other, then the so-called "naïve" approaches can be used [3], [26] to relax the computation requirements.

# Reinforcement Learning and Balancing Exploration vs. Exploitation

- A common dilemma when learning is the following: should I use (exploit) what I know or should I try (exploring) something potentially better? This is commonly known as balancing exploration vs. exploitation.
- In the next two slides we look into two common methods used to address this dilemma. Some other methods can be found in [28].

# The $\epsilon$-greedy method

- The simplest method to balance exploration vs. exploitation [29].
- Randomly explores the different methods with probability $\epsilon$ and uses the method with the highest average throughput with probability 1-$\epsilon$.
- $\epsilon$ -greedy can very effective [30], [31] when the appropriate $\epsilon$ is chosen for the operating scenario.
  - However, at times it may be inefficient and suboptimal.
- ***Pros:*** Simple and effective in many cases
- ***Cons:*** Its performance can be inconsistent across scenarios.

# The Gittins Index

- Gittins proved that exploration vs. exploitation can be optimally balanced using a dynamic allocation index-based strategy [32]. This strategy maximizes the total sum of rewards collected over a long-term horizon. The strategy is simply to use the method with the highest Gittins index, which is based on the reward statistics of each method and must be estimated only when those statistics change (i.e., only when each method is used). More information and results for CR can be found at [25], [33]
- ***Pros:*** Long term optimal, only one recalculation of the Gittins index per time step.
- ***Cons:***
  - Performance can suffer in the short term.
  - The calculation of the Gittins index is very mathematically and computationally expensive.
  - For common distributions of the rewards (e.g. normal and Bernoulli) lookup tables [32] and approximations [34] are available and are sufficient for most cases.

# Decision Trees

- A decision tree learning (classification) method, given training examples, constructs a decision tree with a series of questions (attribute tests).
  - Given some attributes, the tree is traversed until the arrival at a leaf that corresponds to the most likely answer (class).
  - Decision trees are better suited for discrete attributes.
  - Methods exist that allow decision trees to handle continuous attributes [35]. Gandetto and C. Regazzoni [36] use a decision for classifying observed signals and [16] uses a decision tree to aid in the case selection of a CBR based CE.
- ***Pros:*** Can present complex decisions in a concise way.
- ***Cons:*** They require extensive training. Extra work is needed to accommodate continuous attributes.

# Further Reading

- Artificial Intelligence [1], machine learning [37], decision making [38], and architectures for CRs [39] .

# References


[1] A. He, K. K. Bae, T. R. Newman, J. Gaeddert, K. Kim, R. Menon, L. Morales-Tirado, J. J. Neel, Y. Zhao, J. H. Reed, and W. H. Tranter, "A Survey of Artificial Intelligence for Cognitive Radios," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 4, pp. 1578–1592, May 2010.

[2] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ: Pearson Education, 2003.

[3] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[4] T. W. Rondeau, "Application of Artificial Intelligence to Wireless Communications," Virginia Tech, 2007.

[5] T. R. Newman, B. A. Barker, A. M. Wyglinski, A. Agah, J. B. Evans, and G. J. Minden, "Cognitive Engine Implementation for Wireless Multicarrier Transceivers," *Wiley Journal on Wireless Communications and Mobile Computing*, vol. 7, no. 9, pp. 1129–1142, 2007.

[6] M. Kamal Hossain and A. Abd El-Saleh, "Cognitive Radio Engine Model Utilizing Soft Fusion Based Genetic Algorithm For Cooperative Spectrum Optimization," *ArXiv e-prints*, Apr. 2013.

[7] Y.-X. Zu, J. Zhou, and C.-C. Zeng, "Cognitive radio resource allocation based on coupled chaotic genetic algorithm," *Chinese Physics B*, vol. 19, no. 11, p. 119501, Nov. 2010.

[8] S. Chantaraskul and K. Moessner, "Implementation of a genetic algorithm-based decision making framework for opportunistic radio," *Communications, IET*, vol. 4, no. 5. pp. 495–506, 2010.

[9] F. Ye, R. Yang, and Y. Li, "Genetic Algorithm Based Spectrum Assignment Model in Cognitive Radio Networks," in *2nd International Conference on Information Engineering and Computer Science*, 2010, pp. 1–4.

[10] Z. Zhao and H. Lai, "A cognitive engine based on case-based reasoning quantum genetic algorithm," *IEEE 14th International Conference on Communication Technology (ICCT)*, pp. 224–228, Nov. 2012.

# References Continued


[11] Y. Yang, H. Jiang, C. Liu, and Z. Lan, "Research on Cognitive Radio Engine Based on Genetic Algorithm and Radial Basis Function Neural Network," *Engineering and Technology (S-CET), 2012 Spring Congress on*, pp. 1–5, May 2012.

[12] N. Zhao, S. Li, and Z. Wu, "Cognitive Radio Engine Design Based on Ant Colony Optimization," *Wireless Personal Communications*, vol. 65, no. 1, pp. 15–24, Feb. 2011.

[13] Z. Zhao, S. Xu, S. Zheng, and J. Shang, "Cognitive Radio Adaptation Using Particle Swarm Optimization," *Wireless Communications and Mobile Computing*, vol. 9, no. 7, pp. 875–881, 2009.

[14] S. K. Pal and S. C. K. Shu, *Foundations of Soft Case-Based Reasoning*. Hoboken, New Jersey: 2004.

[15] A. He, J. Gaeddert, K. Bae, T. R. Newman, J. H. Reed, L. Morales, and C. Park, "Development of a Case-Based Reasoning Cognitive Engine for IEEE 802.22 WRAN Applications," *ACM Mobile Computing and Communications Review*, vol. 13, no. 2, pp. 37–48, 2009.

[16] L. Morales-Tirado, J. E. Suris-Pietri, and J. H. Reed, "A Hybrid Cognitive Engine for Improving Coverage in 3G Wireless Networks," *Communications Workshops, 2009. ICC Workshops 2009. IEEE International Conference on*. pp. 1–5, 2009.

[17] A. Amanna, D. Ali, D. G. Fitch, and J. H. Reed, "Hybrid Experiential-Heuristic Cognitive Radio Engine Architecture and Implementation," *Journal of Computer Networks and Communications*, vol. 2012, pp. 1–15, 2012.

[18] B. Le, T. W. Rondeau, and C. W. Bostian, "Cognitive Radio Realities," *Wiley Journal on Wireless Communications and Mobile Computing*, vol. 7, no. 9, pp. 1037–1048, 2007.

[19] L. Fausett, Ed., *Fundamentals of neural networks: architectures, algorithms, and applications*. Upper Saddle River, NJ, USA: Prentice-Hall, 1994.

[20] K. Tsagkaris, A. Katidiotis, and P. Demestichas, "Neural network-based learning schemes for cognitive radio systems," *Elsevier Journal on Computer Communications, In press*, 2008.

# References Continued


[21] N. Baldo and M. Zorzi, “Learning and Adaptation in Cognitive Radios Using Neural Networks,” in *5th IEEE Consumer Communications and Networking Conference*, 2008, pp. 998–1003.

[22] B. Bojovic, N. Baldo, and P. Dini, “A Neural Network based cognitive engine for IEEE 802.11 WLAN Access Point selection,” *Consumer Communications and Networking Conference (CCNC), 2012 IEEE*. pp. 864–868, 2012.

[23] N. Hu and Y.-D. Yao, “Radio access behavior (RAB) based cognitive radio classification and identification,” *Communications (ICC), 2012 IEEE International Conference on*. pp. 5588–5592, 2012.

[24] H. I. Volos, C. I. Phelps, and R. M. Buehrer, “Physical Layer Cognitive Engine for Multi-Antenna Systems,” in *IEEE Military Communications Conference*, 2008, pp. 1–7.

[25] H. I. Volos and R. M. Buehrer, “Cognitive Engine Design for Link Adaptation: An Application to Multi-Antenna Systems,” *IEEE Transactions on Wireless Communications*, vol. 9, no. 9, pp. 2902–2913, 2010.

[26] H. I. Volos, C. I. Phelps, and R. M. Buehrer, “Physical layer cognitive engine for multi-antenna systems,” in *IEEE Military Communications Conference*, 2008, pp. 1–7.

[27] S. Bourbia, D. Le Guennec, K. Grati, and A. Ghazel, “Statistical decision making method for cognitive radio,” *19th International Conference on Telecommunications (ICT)*, no. Ict, pp. 1–6, Apr. 2012.

[28] W. Jouini, D. Ernst, C. Moy, and J. Palicot, *Multi-armed bandit based policies for cognitive radio’s decision making issues*. 2009, pp. 1–6.

[29] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[30] H. I. Volos and R. M. Buehrer, “Cognitive Radio Engine Training,” *IEEE Transactions on Wireless Communications*, vol. 11, no. 11, pp. 3878–3889, 2012.

# References Continued


[31] H. I. Volos and R. M. Buehrer, “On Balancing Exploration vs. Exploitation in a Cognitive Engine for Multi-Antenna Systems,” in *IEEE Global Telecommunications Conference*, 2009, pp. 1–6.

[32] J. C. Gittins, *Multi-Armed Bandit Allocation Indices*. Wiley, Chichester, NY, 1989.

[33] H. I. Volos and R. M. Buehrer, “On Balancing Exploration vs. Exploitation in a Cognitive Engine for Multi-Antenna Systems,” in *IEEE Global Telecommunications Conference*, 2009, pp. 1–6.

[34] M. Brezzi and T. L. Lai, “Optimal learning and experimentation in bandit problems,” *Journal of Economic Dynamics and Control*, vol. 27, no. 1, pp. 87–108, Nov. 2002.

[35] R. J. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[36] M. Gandetto and C. Regazzoni, “Spectrum sensing: A distributed approach for cognitive terminals,” *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 3, pp. 546–557, 2007.

[37] M. Bkassiny, Y. Li, and S. Jayaweera, “A Survey on Machine-Learning Techniques in Cognitive Radios,” *Communications Surveys & Tutorials, IEEE*, vol. PP, no. 99. pp. 1–24, 2012.

[38] W. Jouini, C. Moy, and J. Palicot, “Decision making for cognitive radio equipment: analysis of the first 10 years of exploration,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, no. 1, p. 26, 2012.

[39] A. Amanna and J. H. Reed, “Survey of cognitive radio architectures,” in *Proceedings of the IEEE SoutheastCon*, 2010, pp. 292–297.

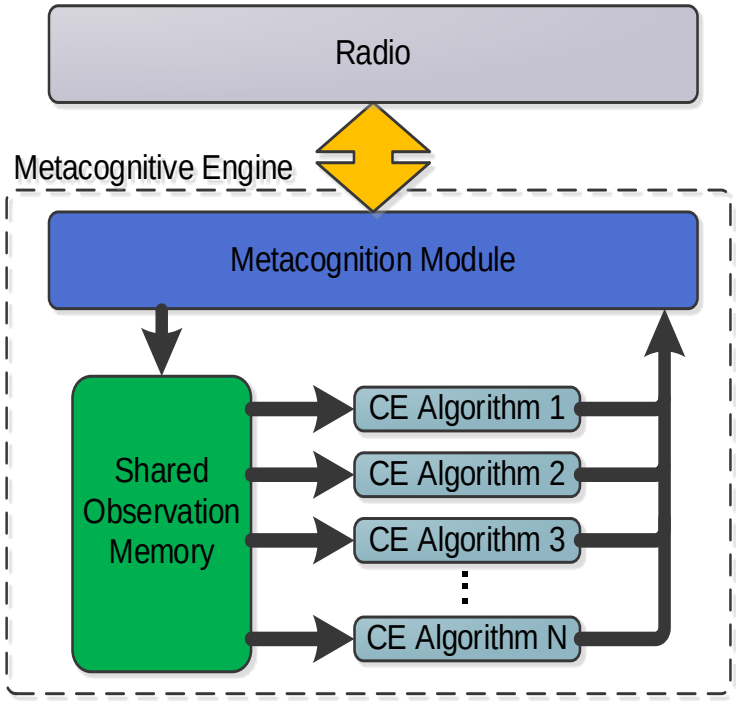
# METACOGNITION AND COGNITIVE ENGINES

# Challenges with single CEs

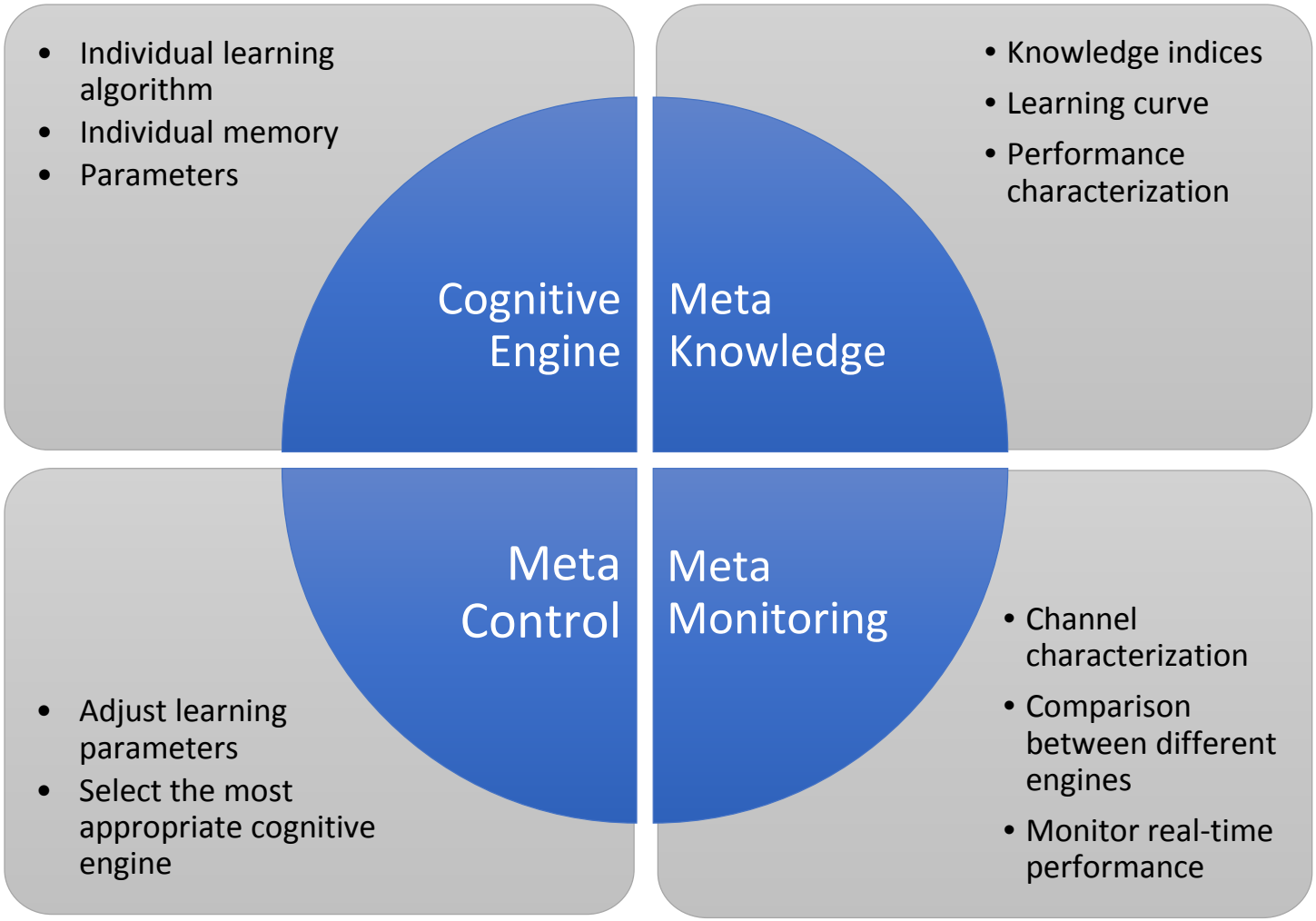
- Most CEs are designed around a 1-2 main algorithms
  - Algorithms have different performance because of their design and settings
    - Makes them appropriate for difference operating scenarios
- Performance predictability
  - A CE that can estimate its own performance is preferred

# The metacognitive solution

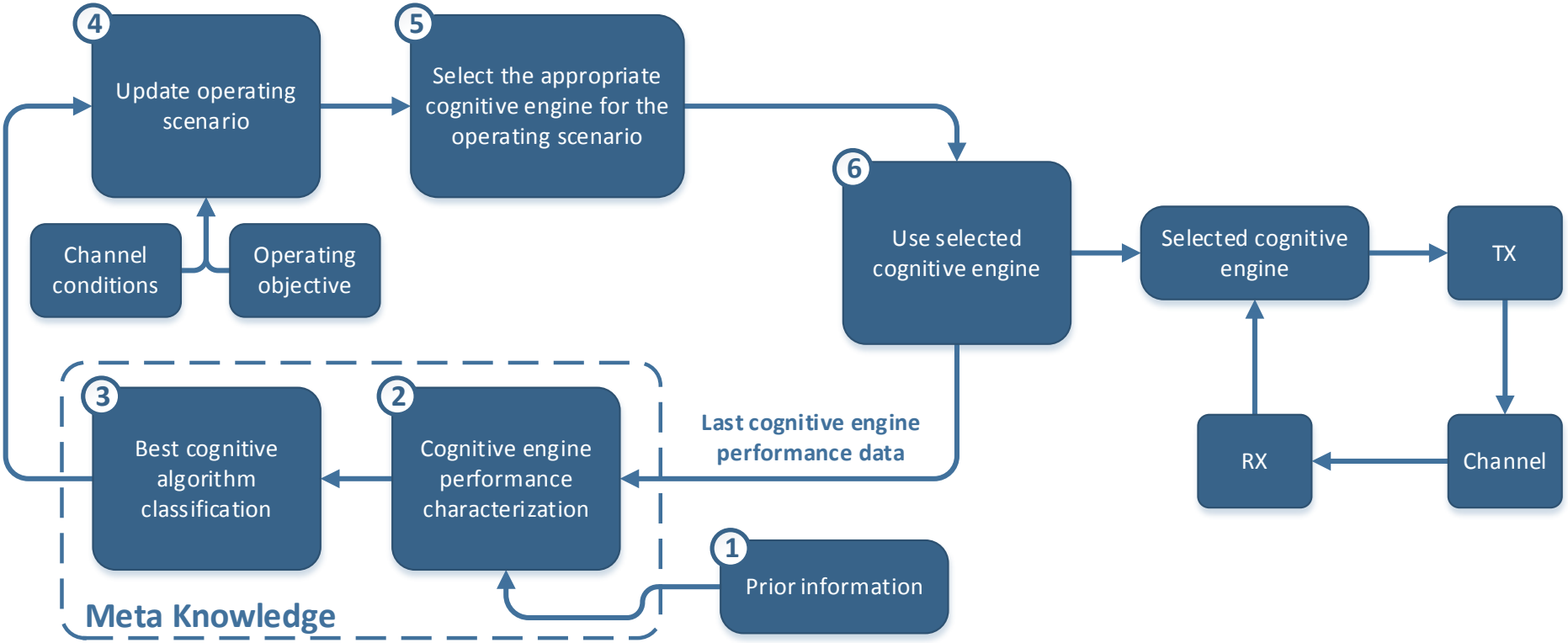
- A meta-CE naturally addresses the single CE challenges
  - It employs multiple algorithms suitable for various scenarios
  - It evaluates each algorithm for the different operating scenarios
  - It recognizes when each algorithm is more applicable



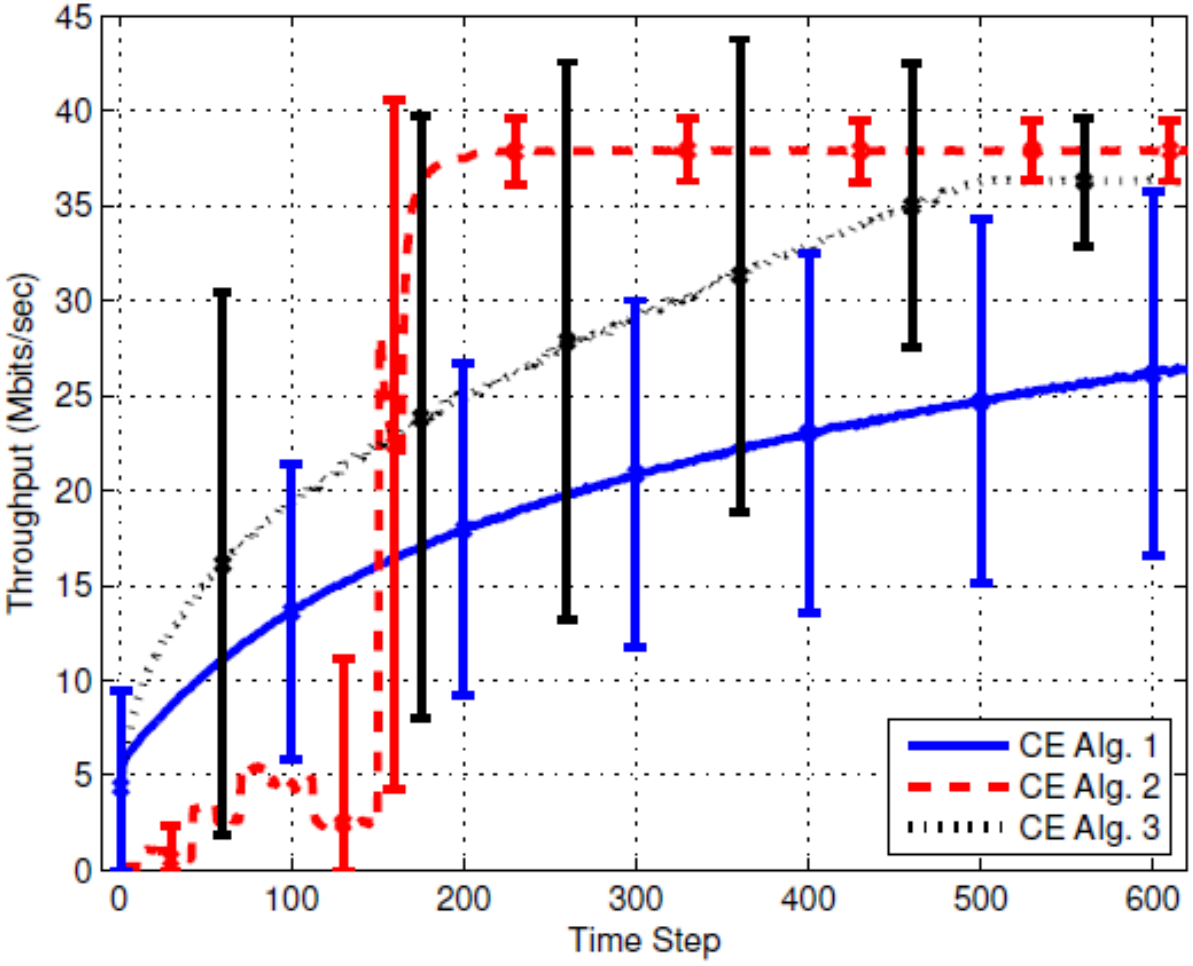
# Metacognitive Components



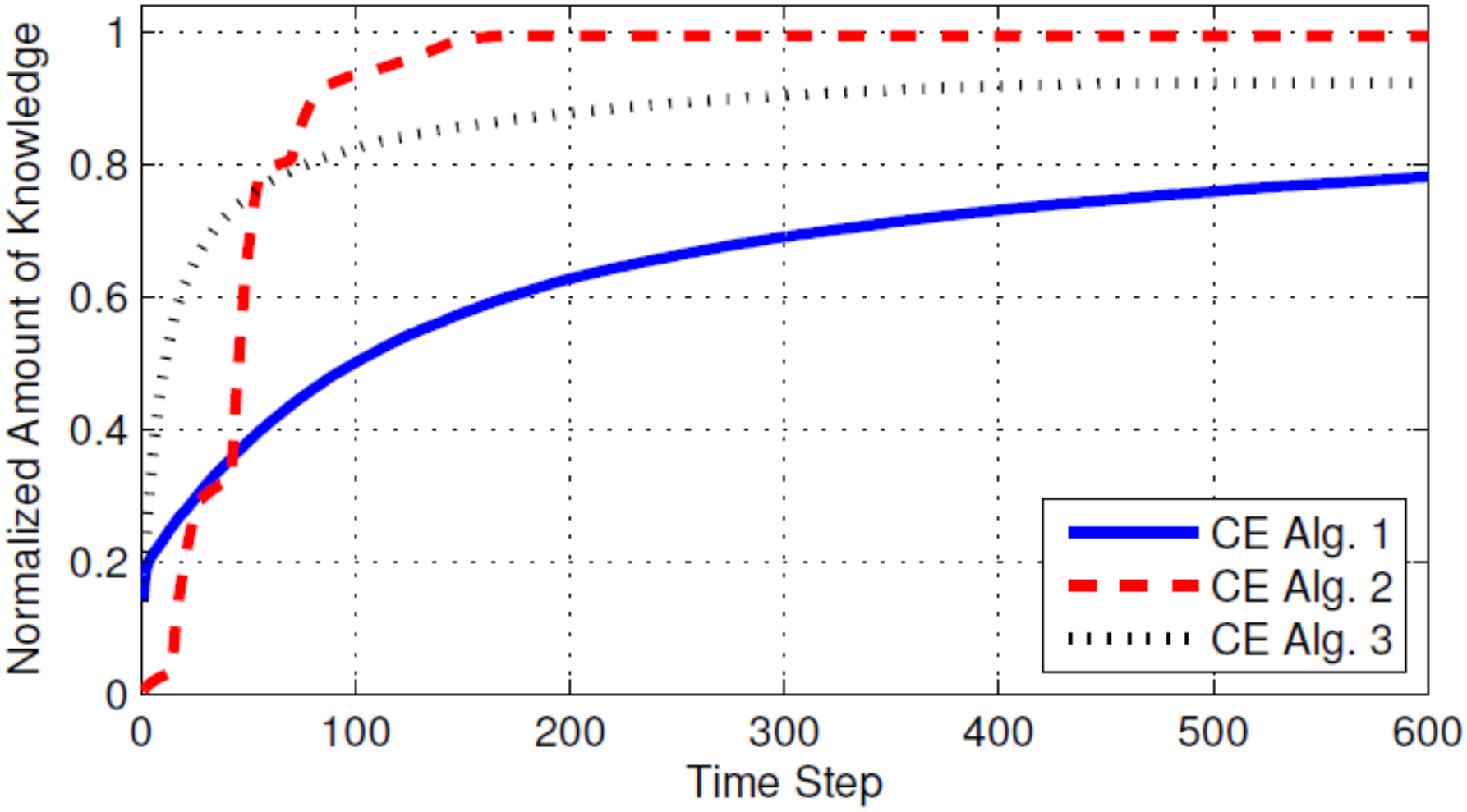
# Metacognitive Engine Operations



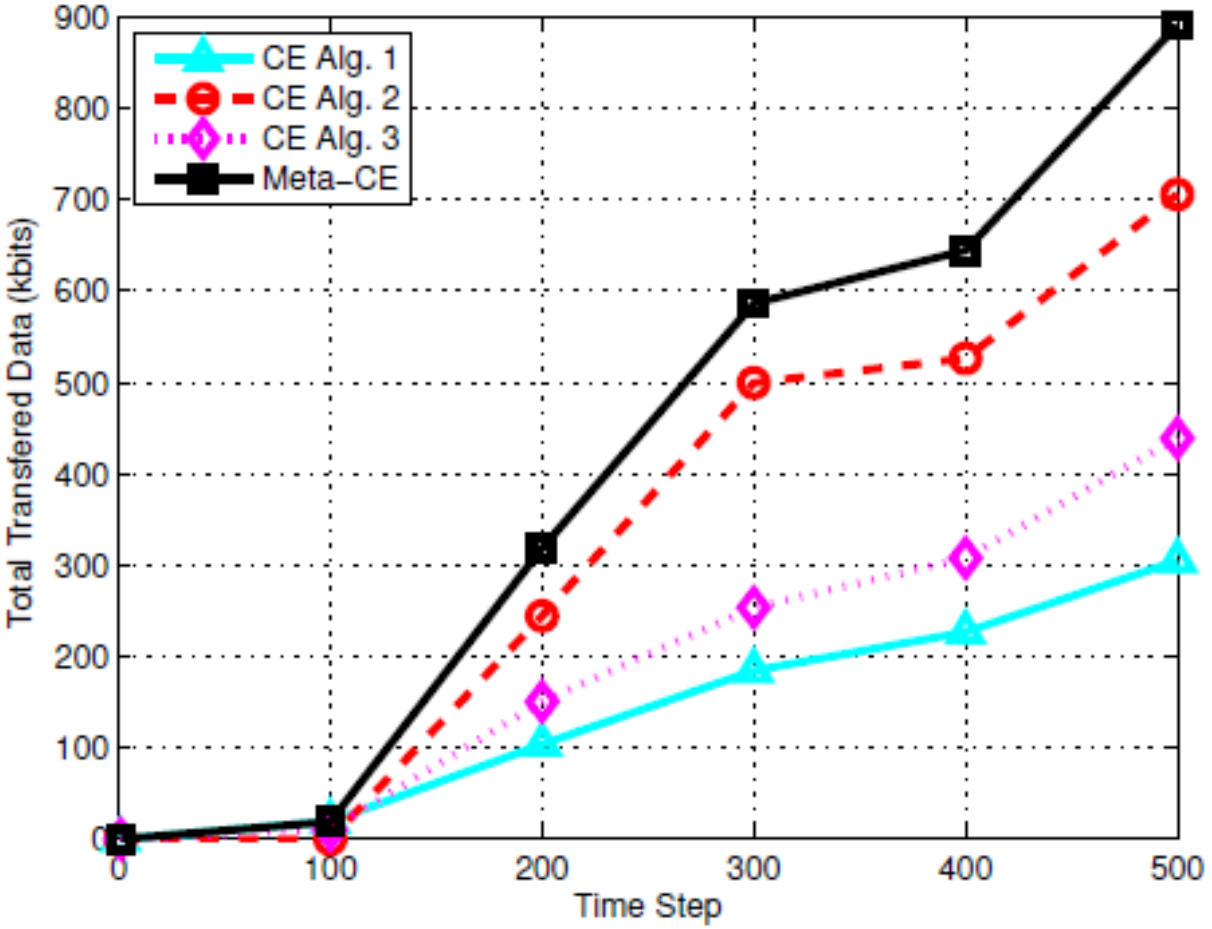
# Learning Curves



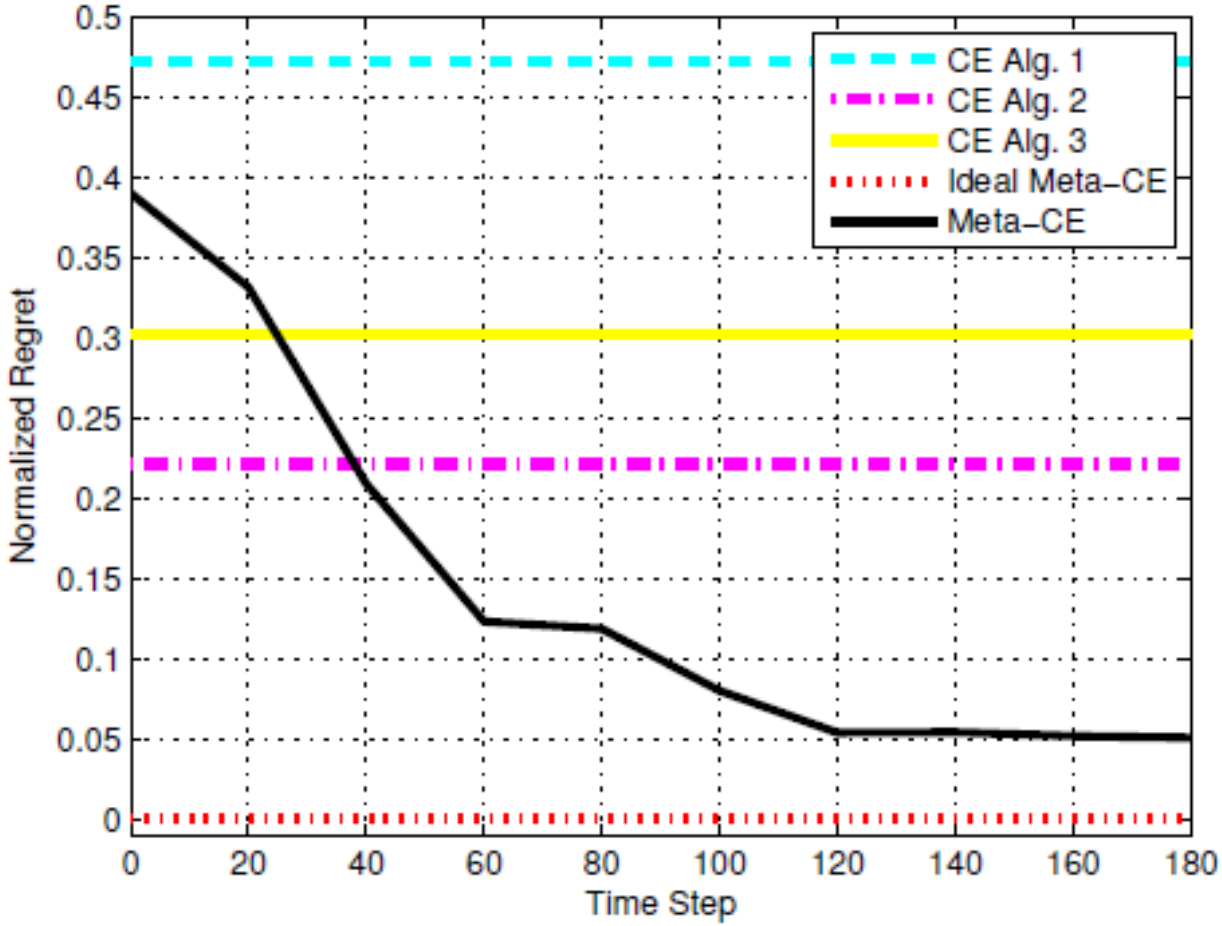
# Estimated Knowledge Learning Curves



# CE Performance (Offline Classification)



# Average Meta-CE Regret (Online classification)



# Meta-CE Bibliography


- H. Asadi, H. Volos, M. Marefat, and T. Bose, “Learning Characterization Framework and Analysis for a Meta-Cognitive Radio Engine,” *Proceedings of SDR-WInnComm,* pp. 132 – 139, Mar. 11 – 13, 2014
- H. Asadi, H. Volos, M. Marefat, and T. Bose, “On Quantifying the Experience Level of a Cognitive Engine,” *Proceedings of SDR-WInnComm*, pp. 9 – 15, Mar. 24– 26, 2015.
- H. Asadi, H. Volos, M. Marefat, and T. Bose, “Metacognitive radio engine design and standardization,” *Accepted in IEEE Journal on Selected Areas in Communications*, 2015.
- H. Asadi, H. Volos, M. Marefat, and T. Bose, “Metacognition and the Next Generation of Cognitive Radio Engines” *Submitted in IEEE communication magazine*, 2015.

# MORE DETAILS AND RESULTS IN METHODS USED

# The Bayes' Rule

$$P(S \mid \mathbf{X}, \mathbf{Y}) = \frac{P(\mathbf{X}, \mathbf{Y} \mid S)P(S)}{P(\mathbf{X}, \mathbf{Y})}$$

**LEGEND:**

$\mathbf{X}$: Channel data ($X_1$: SNR, $X_2$: Eigen spread)

$\mathbf{Y}$: Configuration data ($Y$ Combined modulation and coding distance $d_{min}$)

$S$: 1 if packet successful or 0 if packet dropped

# Bayes' Rule Variations

Naïve Bayes' rule

$$P(\mathbf{X},\mathbf{Y}\,|\,S)=P(X_1\,|\,S)P(X_2\,|\,S)P(Y\,|\,S)$$

Semi Naïve Bayes' (A) rule

$$P(\mathbf{X},\mathbf{Y}\,|\,S)=P(X_1,Y\,|\,S)P(X_2\,|\,S)$$

Semi Naïve Bayes' (B) rule

$$P(\mathbf{X},\mathbf{Y}\,|\,S)=P(X_1,X_2\,|\,S)P(Y\,|\,S)$$

Semi Naïve Bayes' (C) rule

$$P(\mathbf{X},\mathbf{Y}\,|\,S)=P(X_1\,|\,S)P(X_2,Y\,|\,S)$$

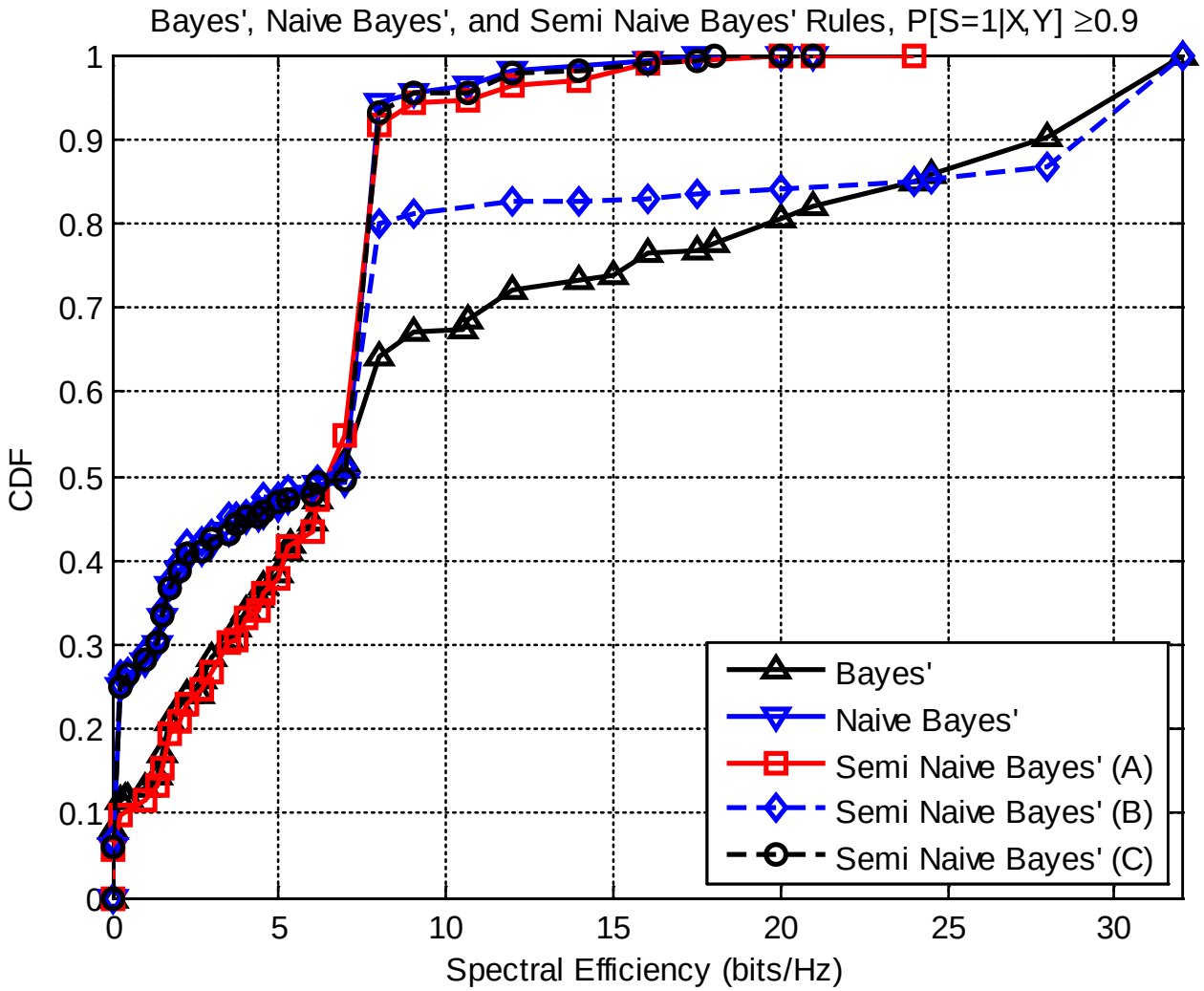
# Test Parameters

- $SNR$: 0 to 50 dB (51 values)
- $log_{10}(Eigen\ spread)$: 0 to 12 (25 values)
- $d_{min}$ 22 values (more details on next slide)
- MIMO schemes: Beamforming, STBC, and V-BLAST

# Tradeoffs between the various approaches

| Model | Mean Spectral Eff.[1] | Parameters required | % Parameters vs Bayes' |
|---|---|---|---|
| Bayes' | 10.5 | 84152 | 100 |
| S.N. Bayes' (A) | 5.9 | 30346 | 36 |
| S.N. Bayes' (B) | 8.6 | 30646 | 36 |
| S.N. Bayes' (C) | 5 | 29254 | 35 |
| Naïve Bayes' | 5 | 28248 | 33 |

[1] $P(S=1|\mathbf{X},\mathbf{Y}) \geq 0.9$


Bayes', Naive Bayes', and Semi Naive Bayes' Rules, P[S=1|X,Y] ≥0.9
CDF
Spectral Efficiency (bits/Hz)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
5
10
15
20
25
30
Bayes'
Naive Bayes'
Semi Naive Bayes' (A)
Semi Naive Bayes' (B)
Semi Naive Bayes' (C)

# Exploration vs. Exploitation

- When the CE has limited knowledge, it has to **choose between** using a configuration that is **proven to work** and a more **promising configuration** with **unknown performance** at the given channel conditions.
- This is the classical problem of **balancing exploration *vs.* exploitation.** *Exploration* refers to trying a configuration that potentially can yield better performance, *exploitation* refers to using a configuration with known performance.

# Goal

- The goal is to balance exploration vs. exploitation.
  - In other words, to minimize the performance cost required for learning/finding the optimal configuration.
- In this work, the performance metric (return) is the achievable capacity of the system.

# The Multi-Armed Bandit Problem

- The exploration vs. exploitation problem is often studied using the Multi-Armed Bandit (MAB) framework.
- The MAB problem assumes that one has to choose a machine $y$ between $K$ machines which yields an unknown reward $R_y$ based on an underlying distribution.
- The goal is to find a policy that maximizes the expected return $V(s_0)$, where $s_0$ is a belief state about the machines' return distributions:

# Expected Total Discounted Reward

$$V(s_0) = E\left\{ \sum_{n=1}^{\infty} \gamma^n R(n) \middle| s(1) = s_0 \right\}$$

where

- *E{· }* is the expectation operator over a given policy starting at state $s_0$
- $\gamma$ a discount factor, $0 < \gamma < 1$
- *R(n)* the return at time index *n*

- Exploration vs. exploitation strategies seek to maximize $V(s_0)$; in this work we consider two key strategies:
  - The epsilon-greedy strategy
  - The Gittins' Index

# The Epsilon-Greedy Strategy

- The epsilon-greedy strategy simply exploits (selects the best known configuration) $1 - \varepsilon$ of the time, and explores (randomly selects a configuration) $\varepsilon$ of the time.
- The benefit of the epsilon-greedy strategy is its simplicity. Its main drawbacks are:
  a) It is not guaranteed to be optimal in a finite time horizon and
  b) it may suffer when the number of configurations is very large.

# The Gittins' Index

- Gittins in [2] showed that the *K*-dimensional MAB problem can be solved by using a dynamic allocation index method that breaks the problem in a series of *K* one-dimensional problems.
- The optimal policy, for each belief state $s_0$ is to use the configuration *y* with the highest index $v_y$:

$$\nu_y(s_0) = \sup_{N>1} \frac{E\left\{\sum_{n=1}^{N-1} \gamma^n R_y(n) \middle| s_y(1) = s_0\right\}}{E\left\{\sum_{n=1}^{N-1} \gamma^n \middle| s_y(1) = s_0\right\}}$$

where $R_y(n)$ is the return of the configuration *y* at the *n*th trial and *N* a stopping time

# Return Distributions

- The Gittins' indices must be derived for the underlying return distributions.
- Gittins' derives and provides tables for the most common return distributions in his book.
- In this work we consider the Normal Return Process (NRP) and the Bernoulli Return Process (BRP)
- The BRP is applicable because the return is either 0 or equal to the capacity of the communication method used (if the packet was successfully delivered)

# Sample Gittins Indices Table, BRP

TABLES

**Table 7. Bernoulli Reward Process Index Values, $a = 0.7$**

| $\beta$ \ $\alpha$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | .6046 | .7358 | .7985 | .8359 | .8612 | .8794 | .8932 | .9041 | .9129 | .9202 |
| 2 | .4118 | .5650 | .6520 | .7088 | .7489 | .7790 | .8024 | .8213 | .8367 | .8496 |
| 3 | .3075 | .4546 | .5472 | .6121 | .6599 | .6970 | .7265 | .7506 | .7708 | .7878 |
| 4 | .2434 | .3789 | .4701 | .5370 | .5887 | .6296 | .6627 | .6904 | .7137 | .7337 |
| 5 | .2005 | .3237 | .4116 | .4779 | .5305 | .5734 | .6088 | .6386 | .6640 | .6861 |
| 6 | .1699 | .2822 | .3654 | .4304 | .4825 | .5259 | .5626 | .5938 | .6207 | .6441 |
| 7 | .1471 | .2497 | .3282 | .3911 | .4426 | .4856 | .5225 | .5545 | .5824 | .6068 |
| 8 | .1295 | .2238 | .2978 | .3581 | .4085 | .4511 | .4878 | .5199 | .5483 | .5734 |
| 9 | .1155 | .2025 | .2724 | .3301 | .3791 | .4211 | .4575 | .4894 | .5179 | .5434 |
| 10 | .1042 | .1849 | .2508 | .3062 | .3535 | .3946 | .4306 | .4624 | .4906 | .5162 |
| 11 | .0948 | .1700 | .2324 | .2854 | .3311 | .3712 | .4066 | .4381 | .4663 | .4916 |
| 12 | .0870 | .1573 | .2164 | .2671 | .3114 | .3504 | .3851 | .4161 | .4441 | .4694 |
| 13 | .0803 | .1463 | .2024 | .2510 | .2938 | .3317 | .3657 | .3962 | .4239 | .4490 |
| 14 | .0745 | .1367 | .1901 | .2368 | .2781 | .3149 | .3481 | .3781 | .4054 | .4303 |
| 15 | .0695 | .1283 | .1792 | .2240 | .2639 | .2997 | .3320 | .3615 | .3884 | .4131 |
| 16 | .0651 | .1208 | .1694 | .2125 | .2510 | .2859 | .3175 | .3463 | .3728 | .3971 |
| 17 | .0613 | .1141 | .1606 | .2021 | .2394 | .2733 | .3041 | .3323 | .3583 | .3823 |
| 18 | .0578 | .1081 | .1527 | .1927 | .2288 | .2617 | .2918 | .3194 | .3449 | .3686 |
| 19 | .0547 | .1027 | .1455 | .1841 | .2191 | .2510 | .2804 | .3075 | .3325 | .3558 |
| 20 | .0519 | .0978 | .1390 | .1762 | .2101 | .2412 | .2699 | .2964 | .3209 | .3438 |

# Example

| $R$ | $\alpha$ | $\beta$ | $v$ | $vR$ |
|---|---|---|---|---|
| 1 | 1 | 1 | 0.6046 | 0.6046 |
| 2 | 1 | 1 | 0.6046 | 1.2092 |
| **3** | **1** | **1** | **0.6046** | **1.8138** |
| 4 | 2 | 4 | 0.3789 | 1.5156 |
| 5 | 1 | 5 | 0.2005 | 1.0025 |

# Example Continued

Assume that the last attempt failed

| $R$ | $\alpha$ | $\beta$ | $v$ | $vR$ |
|---|---|---|---|---|
| 1 | 1 | 1 | 0.6046 | 0.6046 |
| 2 | 1 | 1 | 0.6046 | 1.2092 |
| 3 | 1 | 2 | 0.4118 | 1.2354 |
| **4** | **2** | **4** | **0.3789** | **1.5156** |
| 5 | 1 | 5 | 0.2005 | 1.0025 |

# Test Specifics

- 3 x 22 = 66 different options
- 22 Modulation and coding combinations: from QPSK 1/8 codec to 256 QAM uncoded
- 3 MIMO schemes: Beamforming, STBC, and V-BLAST

# Notes on the results

- The term “optimal” is used to refer to the maximal abilities of the given radio system.
- The total return is the sum of the achieved capacities achieved at a given number of trials. Dropped packets are counted as zero.
- Comparing the actual total return to the optimal total return we get an estimate on the losses due to learning
  - When learning cost = 0, actual total return = optimal total return

# Results

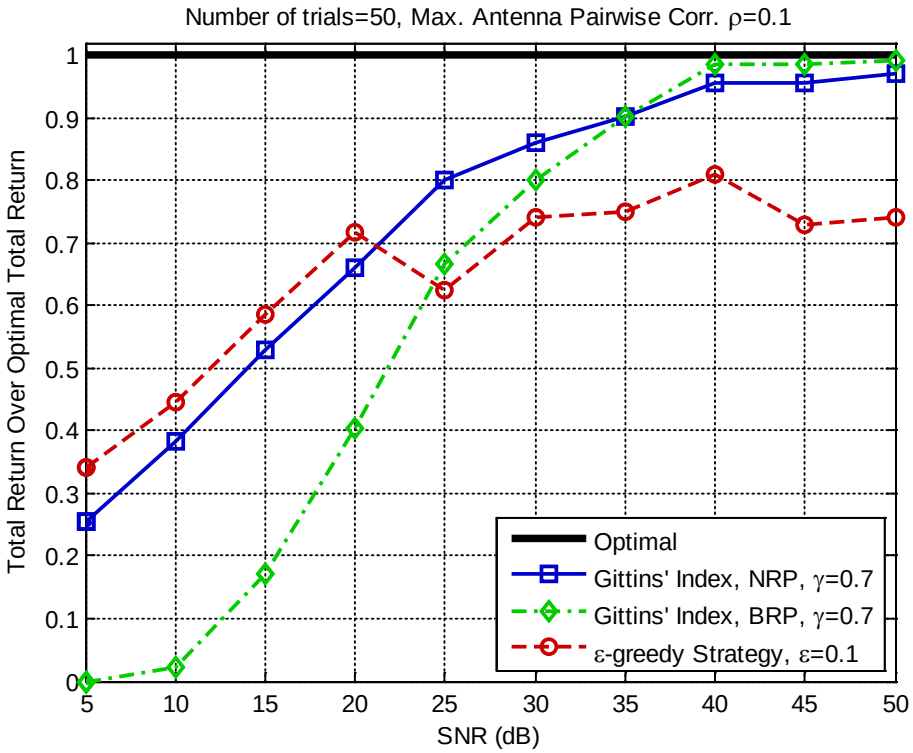
AVERAGE TOTAL RETURN OVER OPTIMAL TOTAL RETURN

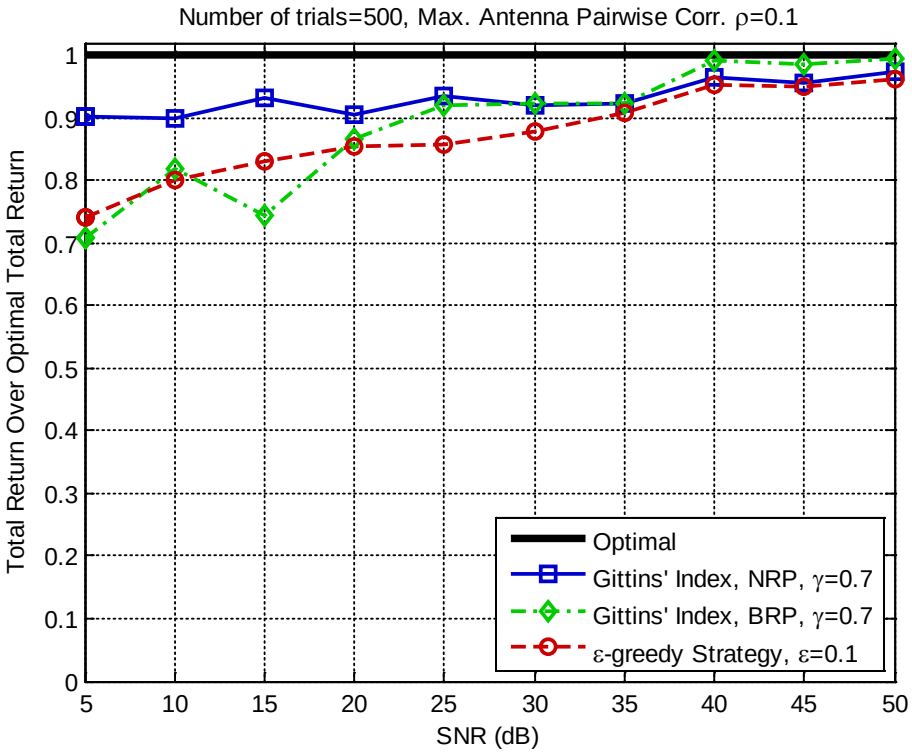
| | Number of Trials Performed | | | | | |
|---|---|---|---|---|---|---|
| | 50 | 50 | 50 | 500 | 500 | 500 |
| | Discount Factor, $\gamma$ | | | | | |
| Method | .5 | .7 | .99 | .5 | .7 | .99 |
| Gittins' Index, NRP | .73 | .73, .70[1] | .72 | .93 | .93, .93[1] | .94 |
| Gittins' Index, BRP | .60 | .59, .65[1] | .56 | .89 | .89, .87[1] | .85 |
| | Exploration Parameter, $\epsilon$ | | | | | |
| Method | .01 | .1 | .2 | .01 | .1 | .2 |
| $\epsilon$-greedy strategy | .53 | .65, .50[1] | .70 | .74 | .87, .87[1] | .87 |

[1]Max. pairwise antenna correlation, $\rho$, equal to .5, .1 otherwise

- The Gittins' index methods outperform the epsilon-greedy strategy
- The epsilon-greedy strategy has very good short term performance

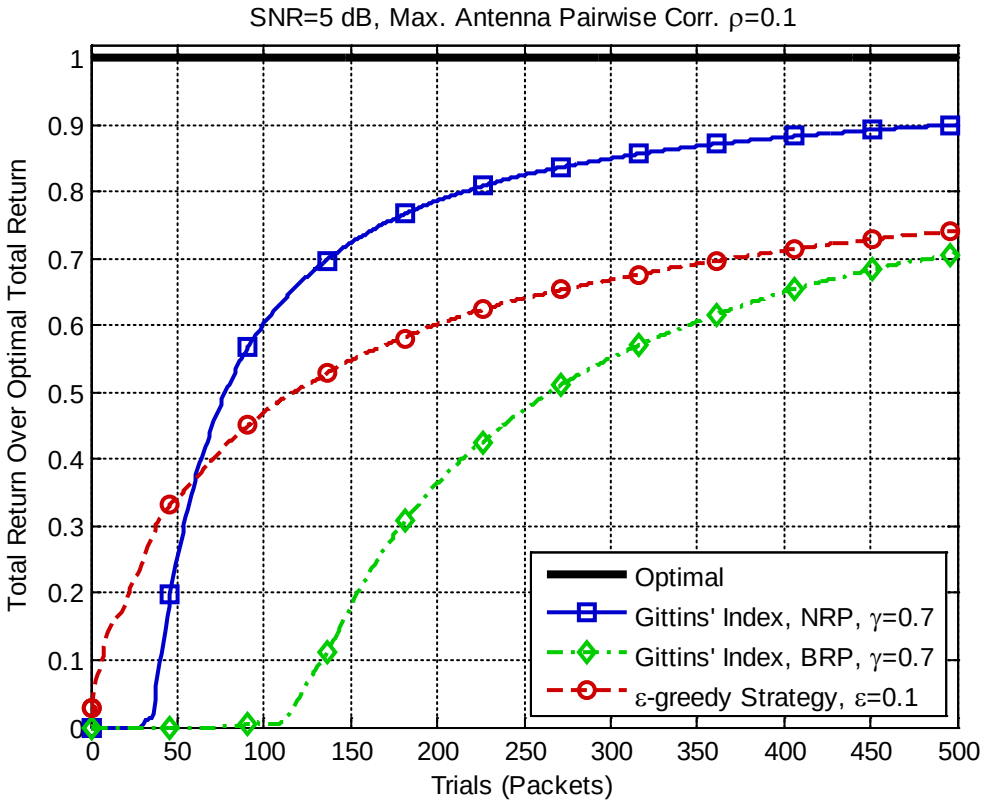
# Total Return vs. SNR,



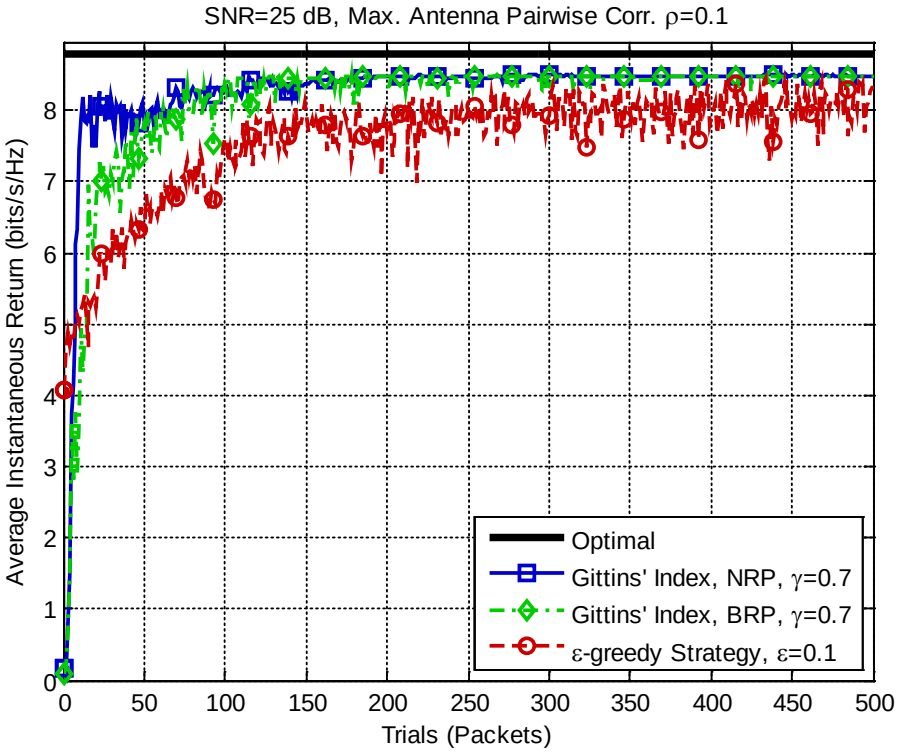

- The same trends as the previous table are observed
- The Gittins' index with NRP was found to have the best overall performance
- Performance at lower SNRs is reduced because fewer options work

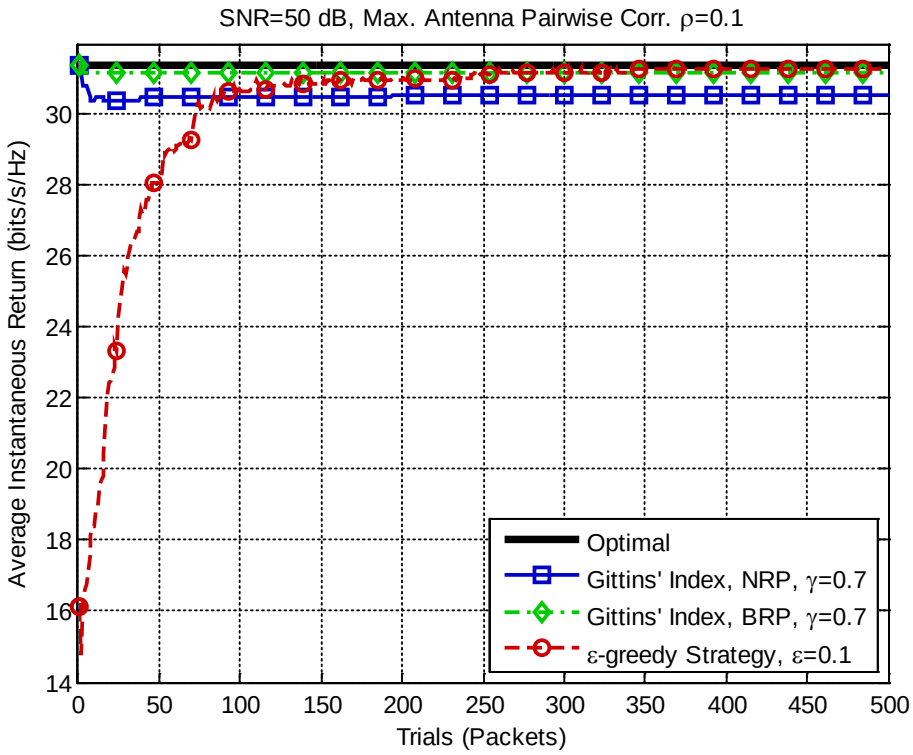
# Total Return vs. Trials




- The epsilon- greedy method has better short-term results (<50 trials).
- The Gittins' index NRP method has better long term performance
- The Gittins' index BRP method has bad initial performance, but it rapidly improves

# Average Instantaneous Return Vs Trials




- after 150 trials all the methods achieve a return very close to the return achieved after 500 trials

At a medium SNR (left figure) the two Gittins' index methods perform the same after 100 trials.

# Robust Training Algorithm (RoTA) Definitions

**Training Method**

A method that can potentially meet the minimum performance requirement ( $\theta_{T_u} \geq \theta_{min} > \theta_{T_l}$).

**Offsetting Method**

A method that exceeds the minimum performance requirement ($\theta_{O_l} > \theta_{min}$).

# RoTA Pseudocode

**Input:** $K$ methods, $\theta_{min}$, $C_{min}$, and $N_w$
**Result:** Runs until $\mathbb{T} = \emptyset$

Populate $\mathbb{T}$ and $\mathbb{O}$
$\alpha_w \leftarrow 0$, $\beta_w \leftarrow= 0$
$\boldsymbol{\alpha} \leftarrow \{\}$, $\boldsymbol{\beta} \leftarrow= \{\}$
**while** $\mathbb{T} \neq \emptyset$ **do**
    **while** $\mathbb{O} = \emptyset$ **do**
        find the training method with $\max\{\theta_{T_l}\}$
        **for** $i = 1$ to $N_w$ **do**
            use the training method
            **if** $\theta_{T_u} < \theta_{min}$ **or** $\theta_{T_l} \geq \theta_{min}$ **then**
                break for
            **end if**
        **end for**
        update $\mathbb{T}$ and $\mathbb{O}$
    **end while**
    **if** empty training list **then**
        break while
    **else**
        pick a high $\theta_{O_l}$ offs. method
        randomly pick a tr. method
    **end if**
    **for** $i = 1$ to $N_w$ **do**
        **if** $\frac{a_w}{a_w+b_w+1} \geq \theta_{min}$ **then**
            use the training method
            update $\boldsymbol{\alpha}, \boldsymbol{\beta}, \alpha_w$, and $\beta_w$,
            **if** $\theta_{T_u} < \theta_{min}$ **or** $\theta_{T_l} \geq \theta_{min}$ **then**
                break for
            **end if**
        **else**
            use the offs. method
            update $\boldsymbol{\alpha}, \boldsymbol{\beta}, \alpha_w$, and $\beta_w$,
            **if** $\theta_{O_l} < \theta_{min}$ **then**
                break for
            **end if**
        **end if**
    **end for**
    update $\mathbb{T}$ and $\mathbb{O}$
**end while**

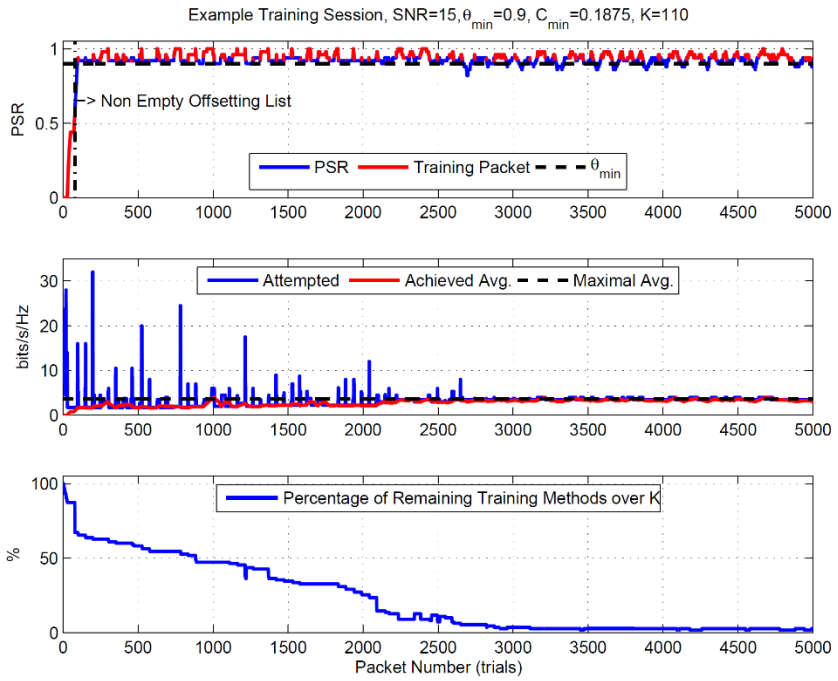
# Minimum Training Window Length

Table: Minimum Training Window Length ($N_w$), $N_T = 1$

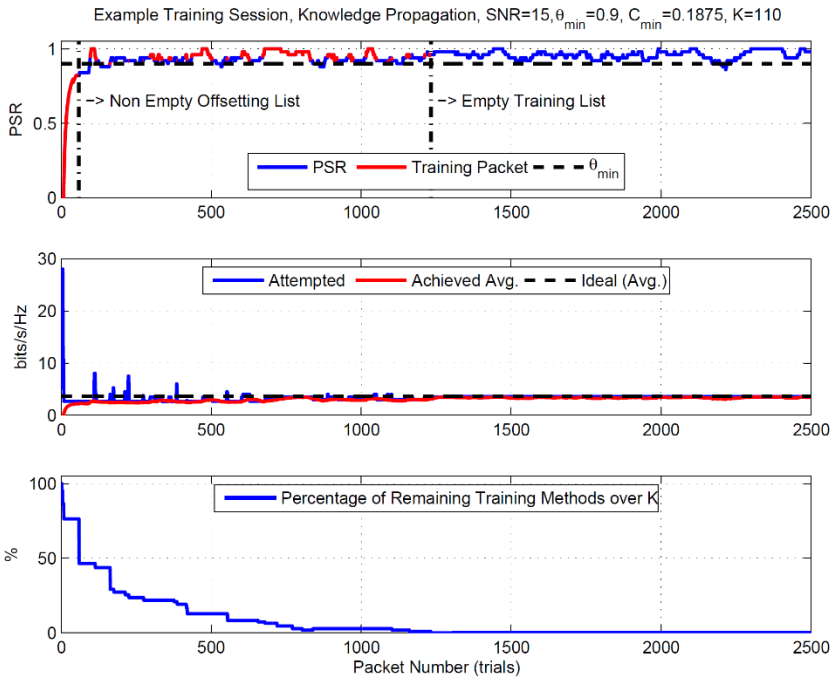
| $\theta_{min}$ | Offsetting Method Lower Bound PSR ($\theta_{O_l}$) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0.55 | 0.60 | 0.65 | 0.70 | 0.75 | 0.80 | 0.85 | 0.9 | 0.95 | 0.999 |
| 0.50 | 11 | 6 | 5 | 4 | 3 | 3 | 3 | 3 | 3 | 3 |
| 0.55 | | 12 | 7 | 5 | 4 | 4 | 3 | 3 | 3 | 3 |
| 0.60 | | | 13 | 7 | 5 | 4 | 4 | 3 | 3 | 3 |
| 0.65 | | | | 14 | 8 | 6 | 5 | 4 | 4 | 3 |
| 0.70 | | | | | 15 | 8 | 6 | 5 | 4 | 4 |
| 0.75 | | | | | | 16 | 9 | 6 | 5 | 5 |
| 0.80 | | | | | | | 17 | 9 | 7 | 6 |
| 0.85 | | | | | | | | 18 | 10 | 7 |
| 0.90 | | | | | | | | | 19 | 11 |
| 0.95 | | | | | | | | | | 21 |

- The smaller the difference between the target and offsetting PSR, the longer the training window needs to be

# Results



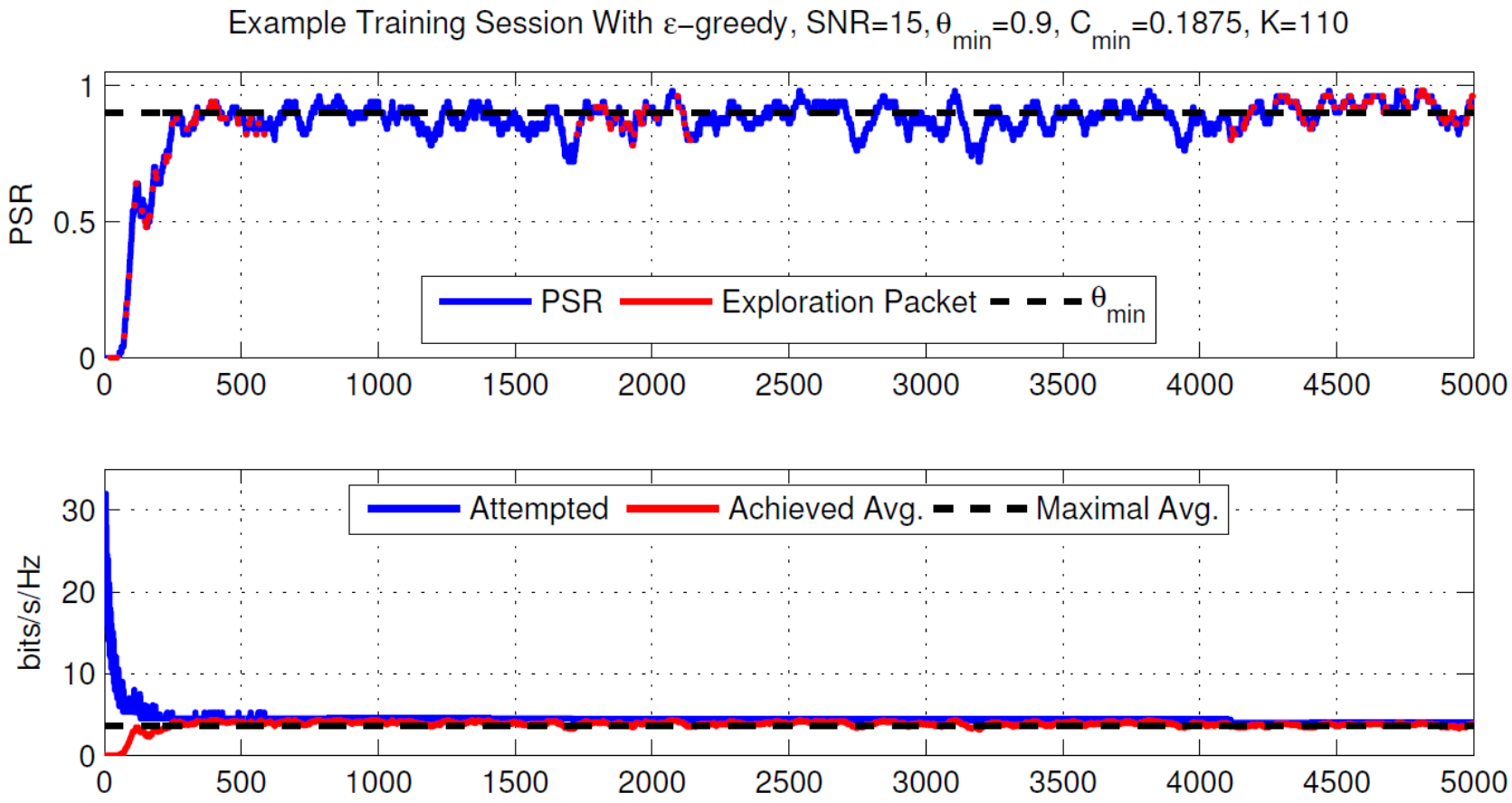

(a) Without Knowledge Propagation




(b) With Knowledge Propagation

- Once an offsetting method is found, the PSR stays close to the target
- New methods are attempted with min. impact to performance
- Knowledge propagation speeds up the operation

# Results


Example Training Session With ε–greedy, SNR=15, θmin=0.9, Cmin=0.1875, K=110
PSR
1
0.5
0
PSR
Exploration Packet
θmin
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
bits/s/Hz
30
20
10
0
Attempted
Achieved Avg.
Maximal Avg.
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000