A Feature Partitioning Approach to Casebased Reasoning in Cognitive Radios

Daniel Ali
Bradley Department of Electrical and Computer Engineering
Virginia Tech
danielali@vt.edu

Jung-Min ”Jerry” Park
Bradley Department of Electrical and Computer Engineering
Virginia Tech
jungmin@vt.edu

Ashwin Amanna
Bradley Department of Electrical and Computer Engineering
Virginia Tech
aamanna@vt.edu

***Abstract*—Cognitive radios have applied various forms of artificial intelligence (AI) to wireless systems in order to solve the complex problems presented by proper link management, network traffic balance, and system efficiency. Case-based reasoning (CBR) has seen attention as a prospective avenue for storing and organizing past information in order to allow the cognitive engine to learn from previous experience. CBR uses past information and observed outcome to form empirical relationships that may be difficult to model using theory. As wireless systems become more complex and more tightly time constrained, scalability becomes an apparent concern to store large amounts of information over multiple dimensions. This paper presents a quickly accessible data structure designed to reduce access time several orders of magnitude as opposed to traditional similarity calculation methods. A framework is presented for case representation, which provides the core of useful information contained within a case. By grouping possible similarity dimension values into distinct partitions called *buckets*, we develop a data structure with constant ($\mathcal{O}(1)$) access time.**

I. Introduction

Cognitive Radio (CR) was first introduced as a technique to adapt a radio to its users' needs, introduced by Mitola [1]. Since then, software defined radio (SDR) technology has enabled a wide variety of CR applications with different objectives in mind.

With a diverse set of applications, CR research has catapulted into an active field of study and seen many theoretic and real world applications. Cognitive Engines (CEs) are the main driver behind decision making in a CR system. CEs are charged with managing more layers of the network stack, the number of parameters and their possible values grows the possible case size exponentially. For example, consider a simple example of all possible parameter values as 4 vectors, each able to take on one of 10 possible values. This results in $10^4 = 10000$ possible combinations for a simple scenario. Allowing these possible values to be on a continuous dimension instead of a discrete one and the problem quickly gains additional complexity. As a CR grows in number of system responsibilities, the possible states of the radio grow even faster.

Traditional Casebased Reasoning (CBR) research advocates an aggressive casebase pruning module [2] [3]. This is an attempt to effectively limit the size of the casebase in order to reduce the access time of information as a whole. This can be unnecessary as memory is becoming less of a concern and processing time remains a more precious resource, especially in the case of tightly timed networks such as IEEE 802.22 [4]. The strength in casebase reasoning is its ability to store past empirical knowledge covering the relevant dimensional space that provides the best performance. This provides motivation of a proper design for a scalable, yet efficient data structure for these relevant dimensions.

A critically important aspect of proper CBR usage is the design of the case. Accurate and relevant information should be contained within the case and a framework is presented from [5] that provides adequate utility subspaces to accurately define parameters and their meaning to the system as a whole. Each space is considered a partition of its own with multiple vectors and can be mapped into other spaces to provide common grounds for relative comparison, such as usefulness (i.e. utility) and similarity (e.g. Euclidean distance).

This paper's contributions can be summarized as follows. First a renewed look at casebased reasoning is presented in the context of communications parameters as presented in [5]. Using these parameters this paper also contributes a new case design that separates the aspects to be considered in case similarity with the contents of the case. The parameters of this similarity dimension of the case design is then bucketed to provide a groundwork for the development of a casebased data structure. Finally, this data structure is presented and reduces the correlation between casebase size and access time as well as preserve good situational descriptions and orderly case retrieval. This is shown through several simulations.

II. Casebased Reasoning for CR

When CBR was first introduced to CR it was under the guise of Case-base Decision Theory (CBDT), which has a natural correlation to CBR [6]. Here we consider the application of CBDT to be synonymous with CBR. CBR incorporates a group of case manipulation mechanisms to drive its reasoning including case representation and indexing, case retrieval, case adaptation, and case-base maintenance [2]. These case manipulation techniques work on the information available in the cases they act on, such as similarity for retrieval, or fitness for projection. In this section, we will lay out the ground work

CROWNCOM 2013, July 08-10, Washington DC, United States
Copyright © 2013 ICST
DOI 10.4108/icst.crowncom.2013.252035

of basic wireless communication for structured storage in these cases.

A. Framework and Parameter Definitions

From [5] we assume that there are three main areas of wireless situations which should be considered from a CE point of view. There are the adjustable parameters which the CE can control, such as power, bandwidth allocation mechanisms, scheduling algorithms, antenna tilt, etc. These may differ with each specific system application and is assumed to be defined in a vector $\boldsymbol{\theta}$. Those things which the CE has absolutely no control over, such as number of users, pathloss, the type of fading, etc. are stored in a vector $\boldsymbol{\phi}$. These two interact to produce measurable performance metrics, such as throughput, QoS requirements satisfied, average number of users blocked etc. defined as $\boldsymbol{\beta}$. The three interact with the channel f as such in Equation 1.

$$\boldsymbol{\beta} = f(\boldsymbol{\theta}, \boldsymbol{\phi}) \tag{1}$$

Typically f is very difficult to model accurately due to mobile users, changing environments, user demands, and radio capabilities, all of which can be dynamic in any given system. Equation 1 provides a generalized and fundamental view of various wireless scenarios which facilitates the understanding of how this data structure can be applied. The advantage of CBR is the ability to store sufficient empirical information on f such that it does not need to adapt a case or create one anew. More in depth information on these parameter definitions can be found in [5].

B. Utility and Fitness

Throughput is an obvious metric for most wireless communications systems but is insufficient to describe the effective utilization of $\boldsymbol{\theta}$ parameters within a system. If the radio places high importance on battery life, throughput may suffer from lower transmission powers, however, overall utilization would be higher than simply trying to optimize throughput. By mapping the important aspects of the system into a common space (a utility space), we are able to accurately compare the usefulness of not only measurable data rate performance, but any aspect of the system that may provide an overall trade off. Here we use the utility and fitness functions defined in [7] and [3]. The utility function q is defined as

$$q(x, \dot{x}; \eta, \sigma) = \frac{1}{2}\left\{1 + \tanh\left[\log\left(\frac{x}{\dot{x}}\right) - \eta\right]\sigma\right\} \tag{2}$$

where the threshold parameter η and the spread parameter σ are chosen such that q reaches 0.95 when the sampled metric, x, reaches the parameter's upper goal, $\dot{x}$. Conversely q is 0.05 when the metric is two decades away from this goal. This function is monotonically increasing and $0 < q < 1$ which can be beneficial to hill climbing approaches. We use this function for utility as it is generalized enough to accurately represent any aspect of the system, regardless of the underlying values they may represent. This function allows us to seamlessly compare decibel to Kbps strictly by how much each contributes to the overall efficiency of the system.

TABLE I
MODIFIED CASE REPRESENTATION

Case Contents	CBDT Attribute
$\hat{u}$	Problem
$\hat{\theta}$	Action
$\boldsymbol{u}$	Result
$\boldsymbol{\delta}$	Similarity

To quantify all utilities together, a global utility, or *fitness* is used to describe how well the utilities within a case are doing as a whole. These utilities usually have weights associated with them and are included in this fitness calculation, known as the production function shown in Equation 3. Where ω_k is the individual weight associated with the given utility u_k.

$$u_{global} = \prod_k u_k^{\omega_k} \tag{3}$$

C. Case Representation

Traditionally, cases contain three main pieces of information: the problem, a solution, and that solution's result. The problem describes the reason the CE was invoked in the first place, meaning that some utility measurement was unsatisfactory and that the system must correct for it. By viewing this as one or many dimensions of $\boldsymbol{u}$, a CE can use this information to identify what aspect of the system has become unsatisfactory. For example if the utility measurement of throughput has dropped below 50%, the CE was invoked and created the case. This is defined as a subset of $\boldsymbol{u}$ such that $\hat{u} \subseteq \boldsymbol{u}$.

The action represents a subset of $\boldsymbol{\theta}$ which contain new values for adjustable parameters in the system to counteract the problem of the case. Through the action, defined as $\hat{\theta} \subseteq \boldsymbol{\theta}$, the CE can provide an updated transmission power, different modulation schemes, or recommend a new adaptation algorithm. Finally, the result is considered as the state of all elements in $\boldsymbol{u}$. This provides the CE with fresh utilization information on throughput, data loss, overhead incurred, etc. to determine how good it's solution really is. Should the same or some other parameter of $\boldsymbol{u}$ be considered unacceptable, then the solution could be considered a "bad" solution and either discarded or retained for future information. A high level description of the case's organization and contents is presented in Table I.

D. Similarity

Similarity is used as the basis for case retrieval in a CBR system. Generally, two usual methods exist to measure a case's similarity; a calculated metric and one that is a measure of the structural representation than a quantified metric [8]. Weighted Euclidean distance is the most common form of similarity calculation as it measure the distance between two points in an n dimensional space. A weight is placed on each dimension and is shown as Equation 4 [2].

$$d_{pq}^{(w)} = d^{(w)}(e_p, e_q) = \left[\sum_{j=1}^{n} w_j^2 (x_{pj} - x_{qj})^2 \right]^{1/2} \tag{4}$$

Where $e_r \subset C$ is the rth case in the set of all cases in the casebase (C) and w is the weight vector for each feature in the case, over n features. x_{pj} is the measure of the jth feature in the respective case p. One of the contributions of this work is to use a well known concept such as Equation 4 while providing a faster lookup time. This is accomplished through bucketing and is discussed in Section III.

Equation 4 assumes that each kth feature should be compared, weighted, and considered in a running average of all features to produce an accurate description of the cases' distances from each other in a n-dimensional case feature-space. This becomes a problem when high granularity is required to differentiate cases. If the delta between some possible feature sample varies much differently than some other feature sample, then a large skew is presented in the distance calculations between the two features.

For example, consider a case with only two features, packet size and overall fitness. Assuming fitness differences are bounded $\in (0, 1)$ using Equation 3 and reasonable deltas of packet sizes (0 - 1800 bytes), we also assume that fitness is weighted to provide 99% of the fitness of the case, versus the 1% of packet size. Even with fitness weighted as heavily as 99% and the maximum difference of fitness (1.0), packet size is the only factor that effectively sways Euclidean distance.

To overcome this downfall of case similarity two ideas are presented. First, we decouple the contents of a case with those aspects that should be considered for similarity. This is defined as $\boldsymbol{\delta} = \{\delta_0, \delta_1, ..., \delta_{N_{\boldsymbol{\delta}}-1}\}$. Secondly, we introduce the concept of bucketing each feature of $\boldsymbol{\delta}$ and partition the subspaces of each feature as a bucket based on granularity. For example δ_0 may be modulation and each modulation would be its own bucket. In a continuous sense, power may be separated by each dB. If it had a max-min difference of 24, and a granularity of 2, then there would be 12 partitions (or buckets as we call them).

III. Methodology

A. Organization By Similarity

Section II-D explains the concept of similarity and how it applies to our case representation. The organization of cases is based on this concept in that it uses each feature of $\boldsymbol{\delta}$ as keys to each level of a tree. To demonstrate this organization, consider the example where the similarity vector is based simply on the transmitter and receiver gain. Given three samples of the two gains, $\{21.4, 30.7\}$, $\{20.6, 31.5\}$, and $\{20.8, 31.3\}$ we can view these cases in a two dimensional space. Each member of $\boldsymbol{\delta}$ is considered a dimension and their partitions are shown in pointer organization in Figure 1. Each transmission bucket contains a separate pointer, each pointing to either NULL if no cases have data values that fall in that range, or to another array. Given that an array exists, it is either another

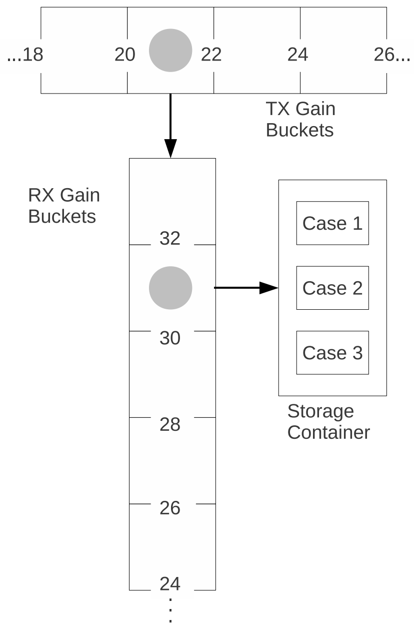

Fig. 1. Gain Bucket Example Organization

dimensional node (receiver gain buckets in Figure 1) or a data container node containing specific case data (storage container in Figure 1). This is typical of tree-like data structures that use levels of hierarchy to index the data via branch nodes and storing the data itself into leaf nodes. By considering each dimension of the similarity vector and finding it's appropriate place in the respective bucket list, a final storage unit stores all cases that are deemed similar by close enough values within their respective similarity dimensions. This storage unit can be presented to the CE for a decision consideration.

For each dimension of $\boldsymbol{\delta}$ we can view these in the data structure as levels of the tree. Each level of the tree has a number of nodes that store their respective pointers to another dimensional node or a leaf node. The organization of these nodes build up a tree structure with the first dimension, δ_0 being the root node and containing pointers to possible children.

B. Machine Interpretation

These buckets and samples that are to be stored in them, need a way to translate a sample's relative position in the list into a machine understandable index. This is possible as long as each dimension of $\boldsymbol{\delta}$ is understood in terms of bucket granularity g_i. The bucket granularity allows the machine to map the buckets described in Section II-D to a memory offset position within the branch node's bucket list. The basic calculation for this index, i, for a dimensional sample, $\hat{\delta_i}$ is expressed in Equation 5.

$$i(\hat{\delta_i}; g_i) = \left\lfloor \frac{\hat{\delta_i} - \min \delta_i}{g_i} \right\rfloor \tag{5}$$

Equation 5 is used for simplicity as it uses the constant g_i to calculate memory offset indices. It is also required that each node be able to calculate the number of children within their own bucket list. This is needed for memory allocations for static array structure. This calculation is presented in terms of δ_i and g_i in Equation 6. g_i can be chosen during design time to be constant or can be expressed as a function of $\hat{\delta_i}$. This is discussed in VI.

$$N_{buckets} = \frac{\max \delta_i - \min \delta_i}{g_i} \tag{6}$$

For most applications, because the number of similarity dimensions (N_δ) will be relatively small, the access time of this organization is dependent on a pre-defined and static number of similarity dimensions. This makes it possible to achieve an access time on the magnitude of $\mathcal{O}(1)$.

IV. Experimentation

Here we describe the advantages of our approach using simulation results. First, we demonstrate that a large casebase experiences performance degradation in large cases and that the bucketed tree effectively reduces this to non-negligible numbers. Next, we demonstrate its usage in an OFDM scenario with a simple CE to show a reasonable percentage of processor time. All simulations are programmed using C++ and run on an Intel Core 2 Duo processor @ 2.53 GHz with 6 GiB of memory in an Ubuntu Linux x64_86 environment. The digital signal processing library *liquid-dsp* is used to simulate the wireless transmissions.

A. Casebase Growth

To demonstrate the power and flexibility of this data structure, an artificial scenario was created to contain 100 similarity dimensions. Each dimension is described in terms of the required elements as described in Section III-B. Each dimension is described in the same fashion, $\max \delta_i = 1000$, $\min \delta_i = 100$, and $g_i = 10$ for $i = 0, 1, ..., N_{\boldsymbol{\delta}} - 1$, where $N_{\boldsymbol{\delta}} = 100$ for these trials. Insertion time and access time are measured for 1000 arbitrary cases in both the tree and the linear implementation. For the linear implementation, only similarity is calculated in these test using Equation 4 and is not ranked, but assumed that the ranking process would be relatively short. This is because the reasoner calling the subroutine would most likely only want a small amount of similar cases, and would adequately rank them during its movement over the list.

Specific arbitrary case information is stored in a case, and is then inserted and searched for using random similarity dimension samples. These samples make up a path through the tree to the case information and is evenly distributed across their dimensional values. While this provides an even distribution across each dimension's sample space more realistic scenarios will have differently distributed probabilities across their dimension. The worst case scenario is used here to show that even without a higher probability in specific areas of each dimension the performance gains can still be seen. This is the worst case in terms of radio performance as each case produces unfamiliar cases, making it more difficult for the CE to converge on a general area where some optimal performance is observed.

B. Wireless Simulation

Next, we simulate a wireless environment with a single transmit/receive pair and a simple reasoner with the goal of link adaptation. By defining a common wireless situation such as link adaptation, the reader can easily identify the usage of the specific parameters defined in this paper. Furthermore, we demonstrate that using this design we can at least show convergence within the bucket scenarios. Because we are bucketing the parameters of similarity a simulation was created to verify that this would provide adequate identifying information for a reasoner to query a case in the casebase. In this wireless scenario, the wireless transmission link is an OFDM signal at 910 MHz using QAM. Packet length was 1024 bytes and the number of sub-carries was 64.

Table II contains a description of each configuration for $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, $\boldsymbol{\phi}$, and $\boldsymbol{\delta}$. These were chosen in order to demonstrate in at least a simple scenario that convergence on a solution is possible with this kind of situation description. While not fully descriptive as a realistic scenario, it is useful to see the convergence on at least one parameter using a randomly generating reasoner. A wider breadth or higher level of MAC parameters such as retransmissions, flow control, etc. can be included in the parameter description in the appropriate place given that the reasoner is charged with their control, however, is not demonstrated here for simplicity.

TABLE II
Parameter Simulation Description

Metric	Value Description
$\boldsymbol{\theta}$	
Transmission Gain	-90 thru 10 dBm
Modulation	BPSK, QPSK, QAM8 - QAM256
Coding	None, Convolution V27 P23-P78, Convolution V28 P23-P78
Packet Size	512 - 2048 bytes
$\boldsymbol{\beta}$	
Throughput	b/s
$\boldsymbol{\phi}$	
Path Loss	0 thru 40 dB
Throughput	0 thru 2000

$\boldsymbol{\delta}$

Similarity Dimension	min	max	Grain
Path Loss	0 dB	40 dB	1 dB
Throughput	0	2000 kbps	200 kbps

The similarity dimension, $\boldsymbol{\delta}$, was chosen to be a simple, yet effective, descriptor in this scenario. Pathloss was varied over time as a square wave to only introduce two levels of pathloss. This could easily be extended to vary pathloss as a sine wave, forcing it to solve solutions for a continuously changing environment.

The utilities chosen is a small set of what could actually be used in a real system shown in Table III. Power consumption and signal strength are chosen for their clear trade-off produced by transmission power. Base-stations might use these differently as well to trade-off reliable transmissions versus overall network throughput. Throughput encompasses many aspects of a wireless transmission, and presents a valuable utility for system performance. Robustness is used to determine how resilient the signal is to fading and/or path loss. When a signal is more robust, it also degrades the capacity of the channel. This degradation will be reflected in throughput as well and thus, does not need its own utility. The weights for the utilities reflect an overly important view of throughput over the others but can change with the goals of the radio. The weights chosen here were chosen arbitrarily to simply reflect a scenario where two aspects were far more important than others.

TABLE III
UTILITY DESCRIPTION

Utility	$\dot{\beta}$	p	Weight
Power Consumption	-65 dBm	-65 / -15 dBm	0.49
Signal Strength	-15 dBm	-15 / -65 dBm	0.01
Throughput	800 kb/s	100 / 800 kb/s	0.49
Robustness	13	13 / 2 Coding	0.01

Path loss may seem like a simple way to measure the environmental effects, as other measures exists such as free space path loss (FSPL), physical distance, multi-path, noise floor, path-loss exponent, however, these measurements are all assumed static and wouldn't provide useful information from case to case. These just represent physical layer properties and as reasoners are tasked with more responsibilities, they can add more information to this, such as number of users, area type (rural, city, etc.), and anything else pertinent to the environment's description.

V. Results

Within the wireless scenario, we allow a random case generator to provide cases for the case base. It generate a random power, tries it in the simulation, calculates the fitness, then stores the information in a case, in the case-base. This is done to show the system is able to converge on an optimal solution, simply by generating random cases, and querying for the most similar cases. The general trend of each simulation was that as the case generator added more cases, the returned set of cases contained cases with higher fitness than the current situation and would subsequently be applied.

A. Organization by Similarity Performance

As a casebase grows, performance of the reasoner should not be noticeably impacted by accessing its learned information. The implementation provided here provides a relatively timely insertion and access time as compared to a traditional list-based casebase. The relative impact of the number of cases can be seen in Figure 2. The linear relationship between the number of cases and access time is clearly shown for a linear data structure, representing traditional approaches for case storage. For the implementation of the similarity tree used in this work, a much more constant access time can be observed providing the basis of justification that information access time can be improved orders of magnitude greater if organized using the bucketed approach. Insertion time is also shown to have relatively little impact on casebase interfacing and is shown here as it provides a slightly different algorithm than access.

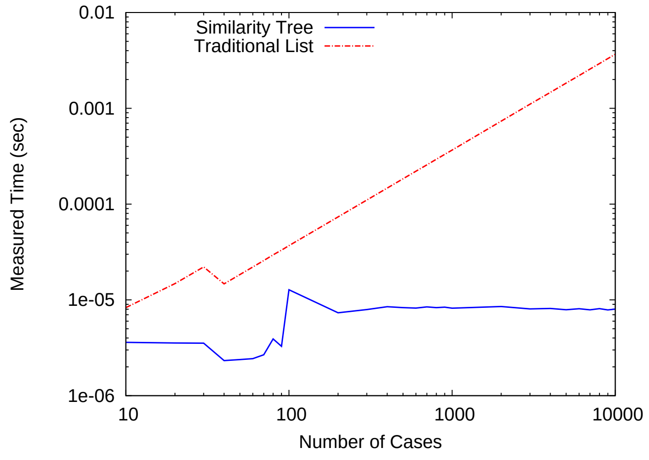

Fig. 2. Average Tree-based and List-based Access Time

It should be noted that this experimentation was done using a controlled simulation to provide the basis of the improvement. More complex reasoning engines may take more time depending on the approach taken to sort through what it considers relative cases. This adds to each measurement some aspect of latency, however, the relationship and improved access time order between the linear and bucketed approach remains the same. Insertion time can also affect the system, however, was measured and did not significantly impact performance. Because this experiment was run using a desktop-level operating system, other influences could have contributed to a non-optimal curve for each trial, such as page faulting or thread scheduling, however, the general trends are preserved in Figure 2.

B. Time Varying Simulation Results

By changing path loss based on a square wave, we introduce two distinct path loss levels in the environment, shown in Figure 3. Each case takes a measurement of an action, records the problem and solution and stores it. By repeatedly forcing a query on the casebase at run-time we are able to see that the data structure eventually converges and produces the best possible solution it could find. It can be seen that even with a simple CE, by retaining simply retaining which cases performed better and then querying them, a fairly adequate trend is visible. Better reasoning mechanisms could know whether the solution is optimal or not.

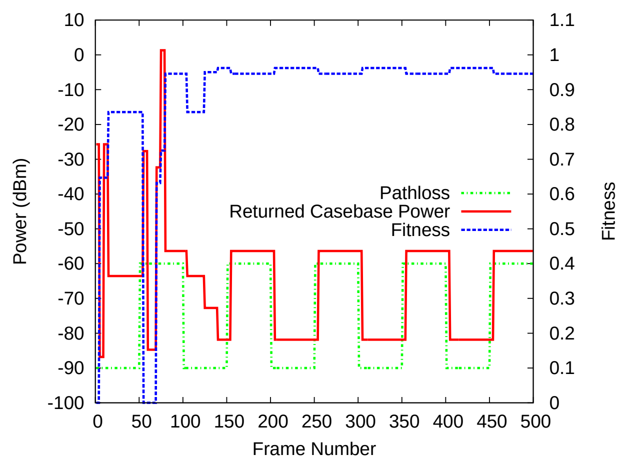

Fig. 3. Time Varying Path Loss Scenario

VI. Limitations and Future Work

One drawback to this kind of bucketed approach is the loss of getting the *k*th nearest neighboring case. Each bucket returns only the cases within the bucket. Should two relatively similar cases fall just on either side of a granularity bound, one or the other is returned, but not both. We believe that by adequately defining the similarity dimensions and its granularity, this problem can go relatively unnoticed. This side effect required more specified scenarios in which to test them in.

To extend this work, a static similarity granularity is used for each dimension of $\boldsymbol{\delta}$, however, this need not be the case. g_i can be considered as a function of δ_i's relative location in the dimension space, allowing for arbitrary bucket sizes that can grow or shrink as it moves through the possible similarity space. While weighting was use in the utility calculation the experiments presented here did not reflect any weighting on similarity. Traditionally, Equation 4 uses a weight to allow one dimension more influence than the others. In terms of bucketing, this can be incorporated into the mapping function from raw parameter space to the dimensional space. These experimentations also used a random case generator to step through the possible parameter space available to the CE. Better approaches are available for use and should be used in parallel with this CBR.

VII. Related Works

He et. al implements case-based reasoning for WRAN 802.22 applications but can be extended to include a variety of other options to optimize [3]. This will lead to the reasoner making more informed decisions without having to sacrifice time due to the size of the case-base. These cases can be used over time and can store a large amount of data for many different scenarios. Wess et. al presents a similar idea, but focuses on how to effectively bridge the gap between structural and surface similarity [9] [10]. The idea of using an n-dimensional hypercube for storing cases in a given search space has been presented previously [11]. These works, however, assumed splitting the search space into subspaces based on the number of cases to be stored in each subspace, allowing an indefinite granularity. Our work however, holds a static value on granularity such that even if a few cases are extremely similar, they do not need multiple layers of structures to distinguish them unless they fall on either side of a dimension's granularity bound.

VIII. Conclusion

A new approach to storing CR case information via a bucketed approach is introduced. By showing a reasonable and extendable framework, we provide the details of how to appropriately classify information by grouping dimensional information together. By first showing the advantages of using a tree-based data structure that aligns itself with this approach, we demonstrate a specific scenario in which it could be used. In both experiments the bucketed, tree-based approach proved to be several orders of magnitude faster in terms of access time. Insertion time was increased relative to traditional methods, however, this increase is negligible in terms of a CE's cognition cycle.

This work was partially sponsored by the NSF through grant CNS-0746925.

References

[1] Joseph Mitola Iii. *An Integrated Agent Architecture for Software Defined Radio*. PhD thesis, 2000.
[2] S.K. Pal and S.C.K. Shiu. *Foundations of soft case-based reasoning*. Wiley series on intelligent systems. John Wiley & Sons, 2004.
[3] An He, Joseph Gaeddert, Kyung Kyoon Bae, Timothy R. Newman, Jeffrey H. Reed, Lizdabel Morales, and Chang-Hyun Park. Development of a case-based reasoning cognitive engine for ieee 802.22 wran applications. *SIGMOBILE Mob. Comput. Commun. Rev.*, 13:37–48, September 2009.
[4] C. Stevenson, G. Chouinard, Zhongding Lei, Wendong Hu, S. Shellhammer, and W. Caldwell. Ieee 802.22: The first cognitive radio wireless regional area network standard. *Communications Magazine, IEEE*, 47(1):130 –138, january 2009.
[5] Joseph D. Gaeddert. *Facilitating Wireless Communications through Intelligent Resource Management on Software-Defined Radios in Dynamic Spectrum Environments*. PhD thesis, Virginia Polytechnic Institute & State University, Blacksburg, VA, January 2011.
[6] Thomas W. Rondeau. *Application of Artificial Intelligence to Wireless Communications*. PhD thesis, Virginia Tech, Blacksburg, VA, 2007.
[7] Youping Zhao, Joseph Gaeddert, Lizdabel Morales, Kyung Bae, Jung-Sun Um, and Jeffrey H. Reed. Development of radio environment map enabled case- and knowledge-based learning algorithms for ieee 802.22 wran cognitive engines. In *Cognitive Radio Oriented Wireless Networks and Communications, 2007. CrownCom 2007. 2nd International Conference on*, pages 44 –49, aug. 2007.
[8] Warren T., Zhang L.Z., and Mount C. Similarity measures for retrieval in case-based reasoning systems. *Applied Artificial Intelligence*, 12(4):267–288, 1998.
[9] Ramon Lopez De Mantaras, Derek Bridge, and David Mcsherry. Case-based reasoning: an overview. *AI Communications*, 10:21–29, 1997.
[10] Stefan Wess, Klaus dieter Althoff, and Guido Derwand. Using k-d trees to improve the retrieval step in case-based reasoning. In *Stefan Wess, Klaus-Dieter Althoff, & M. M. Richter*, pages 167–181. Springer-Verlag, 1993.
[11] Richard H. Stottler, Andrea L. Henke, and James A. King. Rapid retrieval algorithms for case-based reasoning. In *Proceedings of the 11th international joint conference on Artificial intelligence - Volume 1*, pages 233–237, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.