INFONET, GIST
Journal Club

Coexistence Decision Making for Spectrum Sharing among Heterogeneous Wireless Systems

Authors: B. Bahrak, and J.M.J. Park

Publication: IEEE Trans. on W.Com., Mar. 2014
Speaker: Asif Raza

Short summary: In this paper authors tackle spectrum sharing with an objective of enabling coexistence among dissimilar TVWS networks. The sharing problem is defined as multi-objective optimization problem (MOOP). An algorithm to solve the MOOP has also been presented in the paper. Finally the simulation study shows the superiority of the proposed algorithm over existing coexistence decision making algorithms in terms of fairness and percentage of demand served.

I. Introduction

TV whitespace (TVWS) refers to TV channels not used by licensed operators at particular location and particular time. Worldwide efforts are being initiated to utilize TVWS. As a result multiple standards have initiated steps like IEEE 802.22, IEEE 802.11, ECMA-392 etc. It is quite likely that a heterogeneous mix of secondary networks will coexist in TVWS, each with distinct operation parameters (e.g., bandwidth, transmission power, PHY and MAC techniques, etc.). Therefore, IEEE 802.19 WG has presented 802.19.1 standard to enable coexistence among heterogeneous secondary networks operating in the same region. In this paper, authors propose an algorithm called Fair Algorithm for Coexistence decision making in TV whitespace (FACT). The algorithm makes contribution in following directions:

1) *Multiple constraints are used to formulate coexistence decision making algorithm.*
2) *Optimization problem is modeled as energy minimization problem in a modified Boltzmann machine*
3) *Proposed a FACT algorithm to find a Pareto optimal feasible solution*

II. Constraints For Coexistence Decision Making

1) *Contiguous Channels*

The allocation of contiguous channels enables channel aggregation which can result in a throughput increase of more than 60% compared to the best fixed-width configuration.

2) *Interference*

The allocation manager generates interference graph based on a node's location, transmission power, out-of-band emission characteristics, and frequency band. An interference graph provides quantitative information on the adjacent-channel and co-channel interference between each pair of networks within interference range of each other. The interference graph helps to find the minimum frequency separation between two interfering networks and update this value in the interference graph.

3) *Fairness*

Following notion of fairness: spectrum allocation is considered fair if the ratio of the amount of allocated spectrum to the spectrum demand for each of the coexisting networks is the same.

4) *Channel Allocation Invariability*

The algorithm evaluates the tradeoff between the advantages of reallocating a new block of spectrum to a network vs. the costs of reallocation. The two approaches are used for this purpose:

a) A weight is assigned to each constraint to differentiate each one's impact on the reallocation; and
b) A correlation metric between the previous and the current spectrum assignments are defined, and this value is made as large as possible.

The channel allocation invariability constraint prevents triggering decision-making propagation by a small change in the demand of a network and thus can help the system to reduce the channel switching and communication overhead of coexisting networks.

5) *Transmission Scheduling Constraints*

When two networks, *i* and *j*, need to share a channel, a cost value, *Cij* is defined. It represents the cost of scheduling transmission durations on a channel for these networks in a scheduled repetition period. The value of *Cij* is determined based on the following factors.

1) *Channel widths:* when multiple networks need to share the channel, then a swath of spectrum that is sufficiently large to satisfy the largest channel width requirement is allocated. For example an 802.22 network operates on 6 MHz while 802.11af operates on 5 MHz wide channel. If they need to share the channel then a 6 MHz-wide channel is allocated.
2) *MAC strategies*: networks with compatible MAC strategies are preferred to share the channel. It is because networks with incompatible mac strategies, will result in a higher switching delay and packet error rate due to synchronization issues.
3) *Transmission power*: networks with comparable transmission power are preferred to share the channel as large discrepancy in transmission power between two coexisting networks can cause an asymmetric interference relation between the two networks.

III. Problem Formulation of Coexistence Decision Making

The CDM problem is modeled as an energy minimization problem in a modified *Boltzmann machine*.

A. *Boltzmann Machine*

The Boltzmann machine is a stochastic recurrent artificial neural network that combines the principles of simulated annealing with those of neural networks. The value of the neuron (network) *i*'s state, S_i is determined by the output of a thresholding function, f_{out}, as:

$$S_i = f_{out}\left(T_i, \theta_i\right) = \begin{cases} 1 \text{ with prob. } p_i \\ 0 \text{ with prob.} \left(1 - p_i\right) \end{cases} \qquad (1)$$

Where T_i, is the weighted sum of the state values of all the other neurons, $T_i = \sum_j w_{i,j} S_j$ where $w_{i,j}$ is the connection weight between neuron i and neuron j. θ_i, is an appropriate threshold for T_i that controls the value of S_i. Probability $p_i = \frac{1}{1 + e^{-(T_i - \theta_i)/\tau}}$ and τ is temperature. Thus Boltzmann machines have a scalar value associated with each state of the network referred to as the *energy*, E, of the network as,

$$E = -\frac{1}{2} \sum_i \sum_j w_{i,j} S_i S_j + \sum_i S_i \theta_i \qquad (2)$$

B. Problem Formulation

Each neuron is denoted as a triplet (i, j, k). S_{ijk} is the state of neuron (i, j, k), which has two possible values: 0 and 1. $S_{ijk} = 1$ means that the algorithm should assign channel i at time slot j to wireless network k, and $S_{ijk} = 0$ means that no channel should be assigned. Let N is number of coexisting networks, C is number of available channels and T is number of time-slots per period per channel. Let a network 'k' requires n_k number of time-slots. Let f_{kr} defines the minimum frequency separation between networks k and r. Then energy function for each of the constraints is defined as:

1) **Contiguous Channels***:* in Boltzmann machine context, contiguous channels allocation can be expressed as: $S_{ijk} = S_{(i+1)jk}$. Thus its energy minimization is defined like:

$$E_c = \sum_{i=1}^{C-1} \sum_{j=1}^{T} \sum_{k=1}^{N} \left(S_{ijk} - S_{(i+1)jk}\right)^2 \qquad (4)$$

2) **Interference:** since the value of f_{kr} indicates the minimal amount of separation needed to avoid adjacent-channel interference. In the context of the Boltzmann machine, this is equivalent to two neurons, (i, j, k) and (p, j, r) satisfying $|i - p| \geq f_{kr}$. To represent interference constraint using energy function a new variable is defined as:

$$X_{ikpr} = \begin{cases} 1 & \text{if } |i - p| \geq f_{kr} \\ 0 & \text{otherwise} \end{cases} \qquad (5)$$

This variable signifies whether assigning channel i to network k and channel p to network r in the same time slot causes interference or not. The algorithm minimizes following energy function.

$$E_I = \sum_{j=1}^{T} \sum_{i=1}^{C} \sum_{k=1}^{N} \sum_{p=1}^{C} \sum_{r=1}^{N} S_{ijk} S_{pjr} X_{ikpr} \qquad (6)$$

3) Fairness: fairness is maximized if their ratio of spectrum demand over spectrum allocation is maximized ($R_k \leq 1$) or $\left(1 - R_k\right)^2$ is minimized for each network. Here R_k is same for all networks or their variance is minimized,

$$R_k = \frac{\sum_{i=1}^{C}\sum_{j=1}^{T} S_{ijk}}{n_k} \qquad (7)$$

The algorithm needs to minimize following energy function:

$$E_F = \sum_{k=1}^{N}\left(\frac{n_k - \sum_{i=1}^{C}\sum_{j=1}^{T} S_{ijk}}{n_k}\right)^2 \qquad (8)$$

4) Channel Allocation Invariability: in order to prevent triggering a chain reaction of needless spectrum reallocations and mitigate channel switching and communication overhead. The algorithm needs to minimize the following energy function:

$$E_P = \sum_{i=1}^{C}\sum_{j=1}^{T}\sum_{k=1}^{N}\left(S_{ijk} - S'_{ijk}\right)^2 \qquad (9)$$

Where S'_{ijk} defines outcome of previous allocation. Minimizing this function is equal to maximizing the correlation between the current spectrum assignment and the previous spectrum assignment.

5) Transmission Scheduling: if networks *k* and *r* share same channel then algo. adds following energy to total energy fun.

$$E_S = \sum_{i=1}^{C}\sum_{j=1}^{T-1}\sum_{k=1}^{N} C_{kr(k)}\left(S_{ijk} - S_{i(j+1)k}\right)^2 \qquad (10)$$

Where $r(k)$ defines network that shares channel with network '*k*' after time-slot '*j*'.

By using above minimizing functions, the channel allocation problem becomes following MOOP:

$$\underset{\mathbb{S}}{\text{Minimize}} : \{E_S, E_C, E_I, E_F, E_P\} \qquad (11)$$

The solution to MOOP, (11), is unclear as a single solution point that minimizes all objectives simultaneously usually does not exist. Consequently, the idea of Pareto optimality is used. A solution point is Pareto optimal if it is not possible to move from that point and improve at least one objective function without detriment to any other objective function.

Authors adopt weighted-sum method to solve the MOOP and they use positive scalar weights (scheduling, contiguous channel, interference, fairness, channel allocation invariability); $\lambda^S, \lambda^C, \lambda^I, \lambda^F, \lambda^P$, and minimize following energy function:

$$E = \left\{\lambda^S E_S + \lambda^C E_C + \lambda^I E_I + \lambda^F E_F + \lambda^P E_P\right\} \qquad (12)$$

C. Weight selection

The weights define general gauges of relative importance for each objective function. With methods that incorporate a *priori articulation of preferences*, the decision maker indicates preferences about the objectives so that subsequently the algorithm determines a single solution that presumably reflects such preferences. In order to establish the relationship between preferences and weights, authors use a *paired* *comparison method.* Let's assume MOOP contains '*n*' objective functions then this method involves $\frac{n(n-1)}{2}$ paired comparisons. The judgement then results in matrix *A* where each entry $a_{i,j}$ defines how much important is criterion in row *i* when compared with criterion in column *j*? The main diagonal of *A* is all one. Similarly the inverse mapping is done like, $a_{i,j} = a^{-1}_{i,j}$ which means, for example, if $a_{i,j} = 5$ then $a_{j,i} = \frac{1}{5}$. In order to find weights, authors find principle eigenvector of matrix; $v = (v_1, v_2, \cdots, v_n)$ and set the weight of criterion *i* as; $w_i = \frac{v_i}{\sum_{j=1}^{n} v_j}$

Since this method requires objective functions to have similar ranges, thus each obj. fun. is normalized using its average value. Thus equation for weights looks like:

$$\lambda^{\alpha} = \frac{w_{\alpha}}{E^{\alpha}_{avg}} \qquad (13)$$

Where $\alpha \in \{C, S, I, F, P\}$

D. *Problem Formulation using Boltzmann Machine*

In order to formulate problem in 12 in Boltzmann machine, connection weights are thresholds are required. To define them following functions are used:

1) *Kronecker delta function:* $\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$

2) *Euclidean distance tester function:* $\alpha_i(x) = \begin{cases} 1 & \text{if } 1 < i < x \\ 0 & \text{otherwise} \end{cases}$

3) *Unit difference function:* $\gamma_{ij} = \begin{cases} 1 & \text{if } |i - j| = 1 \\ 0 & \text{otherwise} \end{cases}$

The connection weights between two arbitrary neurons, (*i,j,k*) and (*p,q,r*) are defined as: $W_{ijkpqr} = \left\{ 2\lambda^S C_{kr(i,j,k)} \delta_{ip} \gamma_{jq} \delta_{kr} + 2\lambda^C \gamma_{ip} \delta_{jq} \delta_{kr} - \lambda^I \delta_{jq} X_{ipkr} - \frac{2\lambda^F}{n_k^2} \left((1-\delta_{ip}) \| (1-\delta_{jq}) \right) \delta_{kr} \right\}$ and threshold is defined as: $\theta_{ijk} = \left\{ \lambda^S \left(C_{kr(i,j,k)} + C_{kr(i,j-2,k)} \right) + \lambda^C (1 + \alpha_i(C)) + \lambda^P (1 - 2S'_{ijk}) + \lambda^F \left(\frac{-2n_k + 1}{n_k^2} \right) \right\}$. The total input to neuron is thus defined as:

$$T_{ijk} = \sum_{p=1}^{C} \sum_{q=1}^{T} \sum_{r=1}^{N} W_{ijkpqr} S_{pqr} \qquad (14)$$

The thresholding function in Boltzmann machine is defined as:

$$S_{ijk} = f_{out}(T_{ijk}, \theta_{ijk}) = \begin{cases} 1 & \text{with } p_{ijk} \\ 0 & \text{with } (1 - p_{ijk}) \end{cases} \qquad (15)$$

Where $p_{ijk} = \dfrac{1}{1 + e^{-\left(T_{ijk} - \theta_{ijk}\right)/\tau}}$

IV. FACT: A Fair Coexistence Decision Making Algorithm

A. *FACT Algorithm*

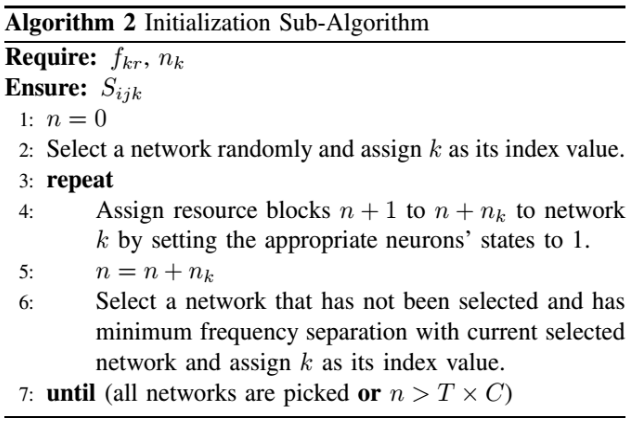
The FACT is an update procedure that runs on above defined Boltzmann machine.

The algorithm starts with initialization step, described in section *B*, that assigns binary values to each neuron's state. The connection weights and threshold values are calculated once and remain fixed. It computes energy function E and repeats until energy function becomes zero or maximum number of iterations (*M*) are reached. Finally the output of algorithm is $BestS_{ijk}$ which represents pareto optimal solution.

The running time of FACT is dominated by complexity of energy computation at each iteration $O\left(N^2C^2T\right)$; defined as: $O\left(MN^2C^2T\right)$

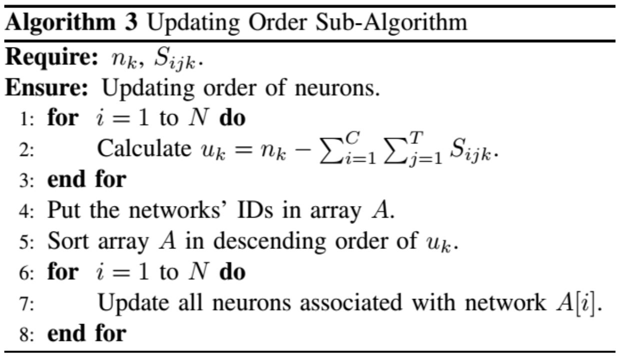
Algorithm 1 FACT Algorithm

```
Require: $f_{kr}$, $n_k$, $\lambda^S$, $\lambda^C$, $\lambda^F$, $\lambda^I$, $\lambda^P$, $S'_{ijk}$, $C_{kr}$
Ensure: $BestS_{ijk}$
 1: Initialize $S_{ijk}$ based on the initialization strategy.
 2: $I = 0$.
 3: Calculate $W_{ijkpqr}$ for all pairs of neurons using Equation (15).
 4: Calculate threshold $\theta_{i,j,k}$ for all neurons using Equation (16).
 5: Compute $E$ using Equation (12).
 6: $MinE = E$.
 7: $BestS_{ijk}$ = $S_{ijk}$.
 8: while ($E \neq 0$ and $I < MaxNumIterations$) do
 9:     repeat
10:         Pick neuron $(i, j, k)$ based on the updating order strategy.
11:         Calculate $T_{ijk}$ using Equation 14.
12:         Calculate $S_{ijk}$ using Equation (15).
13:     until (all neurons are picked)
14:     Compute $E$ using Equation (12).
15:     if $E < MinE$ then
16:         $MinE = E$.
17:         $BestS_{ijk}$ = $S_{ijk}$.
18:     end if
19:     $I = I + 1$
20: end while
```



Algorithm 2 Initialization Sub-Algorithm

```
Require: $f_{kr}$, $n_k$
Ensure: $S_{ijk}$
1: $n = 0$
2: Select a network randomly and assign $k$ as its index value.
3: repeat
4:     Assign resource blocks $n + 1$ to $n + n_k$ to network
       $k$ by setting the appropriate neurons' states to 1.
5:     $n = n + n_k$
6:     Select a network that has not been selected and has
       minimum frequency separation with current selected
       network and assign $k$ as its index value.
7: until (all networks are picked or $n > T \times C$)
```



Algorithm 3 Updating Order Sub-Algorithm

```
Require: $n_k$, $S_{ijk}$.
Ensure: Updating order of neurons.
1: for $i = 1$ to $N$ do
2:     Calculate $u_k = n_k - \sum_{i=1}^{C} \sum_{j=1}^{T} S_{ijk}$.
3: end for
4: Put the networks' IDs in array $A$.
5: Sort array $A$ in descending order of $u_k$.
6: for $i = 1$ to $N$ do
7:     Update all neurons associated with network $A[i]$.
8: end for
```

B. Initialization Strategy

Authors consider all frequency-time blocks, r_{ij} as 1-D array. The algorithm randomly selects a network, *k* and allocates first n_k entries of block to *k* and set neuron *i*'s state as 1. The algorithm then selects another network, say *r* which has min. energy separation with *k* and assigns a number of remaining entries of the array to this network. This process repeats until there is no network left or resources are depleted.

C. Initialization Strategy

This algorithm takes two inputs—the spectrum demand of each network and the current state of its neurons. It first computes how many more resource blocks each network requires (by calculating *uk*), and then it creates a sorted array of network IDs in descending order of each network's *uk* value. For each network, the algorithm updates the state of the neurons according to the order of the networks in the list.

V. Simulation and Results

FACT is compared with two other algorithms named in paper; scheme in [2] and scheme in [3]. The algorithm in [2] selects a network at each step with the minimum quality factor (*Rk*), say network *x*. Then the algorithm tries to find an unoccupied channel for network *x*, and if no unoccupied channel is available, the algorithm searches for a channel occupied by a network with the similar MAC/PHY to the network *x*. If the algorithm finds such a network, it checks to see whether the network supports scheduling, and if it does, it schedules a transmission period for each network on that channel [2]. The complexity of this algorithm is $O\left(N^2CT\right)$. The algorithm in [3] defines Coexistence Value (CV), that measures a network's eligibility for resources. This parameter is computed using the number of nodes in the network, channel utility value that the network can achieve, and a possible regulatory preference [3]. The complexity of this algorithm is $O\left(NCT\right)$.

Simulation Parameters: number of networks =20, number of channels varies from 1 to 20, number of slots per channel = 10, demand of each network (n_k) varies from 5 to 10.

Fig.1 shows how well three algorithms satisfy a network's spectrum demands. Note percentage of demand serviced (PDS); defined as the average of allocated resource ratios for all networks, i.e., $\mathrm{PDS} = 100 \times \left(\frac{1}{N} \sum_{k=1}^{N} R_k \right)$. The difference in PDS performance among the three algorithm are due to the following reasons:

1) FACT maximizes R_k for all networks in the fairness constraint,
2) Scheme of [2] only allows scheduling between networks of the same air interface and the scheme of [3] does not allow transmission scheduling.

The comparison in Fig. 2 uses fairness *F*, as one minus the variance of quality factors of all networks, i.e., $\mathrm{F} = 1 - \frac{1}{N} \sum_{k=1}^{N} \left(R_k - \bar{R} \right)^2$. The FACT performs better than the algorithm of [2] due to the fact that the latter always assigns spectrum to the network with minimum *Rk*, whereas the former allocates the spectrum in a way that makes all *Rk* values equal. The algorithm of [3] shows the worst performance in terms of fairness because it does not consider transmission scheduling.

When the number of available channels is small, the quality factors of most of the networks are close to zero, and as a result we have a high fairness value. By increasing the number of available channels, some networks get more resources than the others, and the fairness decreases. But when the number of available channels is enough for satisfying the demand of most of the networks, the fairness value of all three algorithms increases. The Packet Error Rate (PER) is calculated using formula as: $\mathrm{PER} = \sum_{m=0}^{N} \left[1 - (1 - p)^m \right] f_m(M)$ where *p* is symbol error probability, *M* is random variable that defines number of symbols collide with interference pulse and $f_m(M)$ is prob. Mass fun. of *M*.

Three networks, using 64-QAM modulation, and competing for two channels are considered for PER experiment. The value of *p* for 64-QAM is defined as: $p = 1 - \left(1 - \frac{7}{4} Q\left(\sqrt{\frac{\gamma}{21}} \right) \right)^2$ where γ is SINR and $Q(x)$ is integral of the tail of a normalized Gaussian probability density function. The scheme in [2] outperforms FACT because it uses scheduling among similar networks while FACT considers any kind of networks to perform scheduling. The scheduling among networks with similar MAC/PHY strategies results in less error rate. The scheme in [3] is not considered in PER experiment because it does not support scheduling.

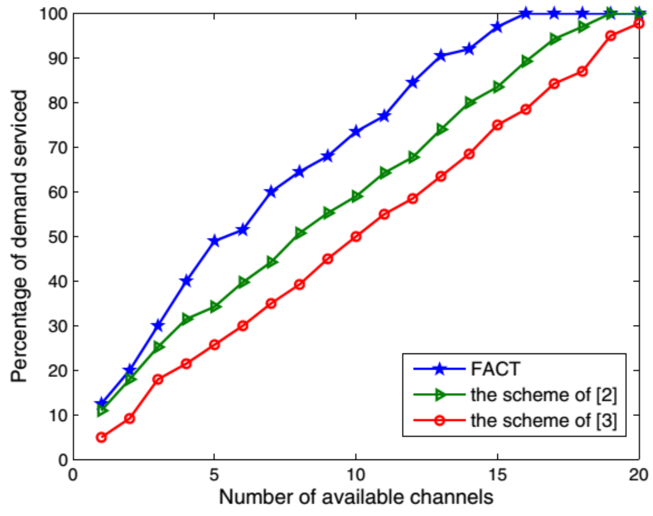


Fig. 1. PDS vs. number of available channels.

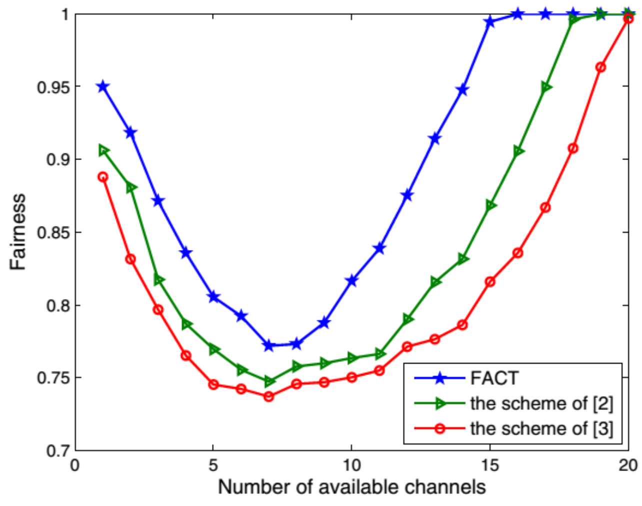


Fig. 2. Fairness vs. number of available channels.

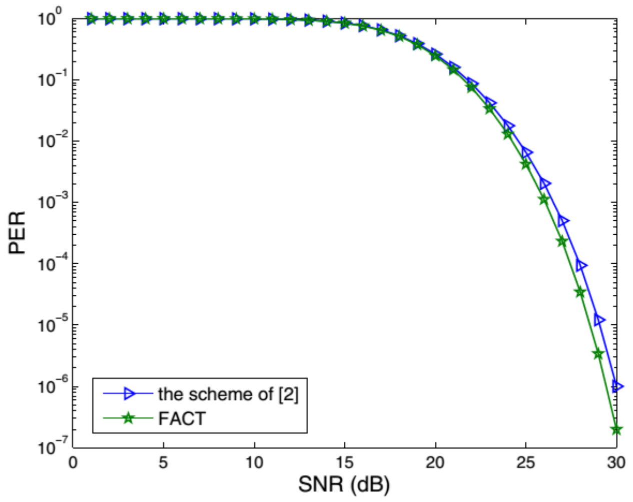


Fig. 3. Packet error rate vs. signal to noise ratio.