AN ALGORITHM FOR FAIRNESS BETWEEN SECONDARY USERS IN COGNITIVE RADIO NETWORK

ABDELLAH IDRISI, *SAID LAKHAL*
Research Computer Sciences Laboratory (LRI)
Computer Sciences Department, Faculty of Sciences
University Mohammed V - Agdal, Rabat
E-mail: 1 idriab@gmail.com, 2 said.lakhal.rech@gmail.com

ABSTRACT

Scheduling Secondary Users (SUs), to exploit the free channels in Cognitive Radio Network, represents one of the major challenges. In this work we propose an algorithm to ensure fairness of service between the secondary users. This fairness is expressed in terms of transfer rates. The algorithm produces a chain containing the order wherein the packets of each SU will be sent. The experimental results provide transfer rates too close to their average. This proves the efficiency of scheduler proposed.

Keywords: Cognitive Radio, Scheduling, Fairness, Standard Deviation.

1. INTRODUCTION

Cognitive Radio Networks (CRN) is an area of research booming. It was initiated by Mitola in [1]. The author has dealt with the problems related specially to the spectra. After the emergence of this field, several areas of research including Artificial Intelligence and Telecommunications... have addressed the issue. Current challenges for the researchers focus on the scarce resources (like spectrum, energy, time and space) and the increasing demands, particularly from secondary users, on unused frequencies. This raises the question of a possible saturation of these frequency bands.

Scheduling in the CRN consists to assign the free channels for SUs so they can transfer their data packets. To do this, we must determine, during a fraction of time, some parameters like free frequencies, the state of the queue of each SU and the procedure for channel allocation.

To achieve these goals, we believe it would be interesting to develop an efficient scheduler to properly control CRN. Our work focuses on a CRN composed of several SUs, only one channel and only one PU.

To obtain Transfer Rate (TR) almost identical between all SUs, we have grouped the SUs which admit the same number of packets to be transmitted in the same group. The proposed method allows placing the Group which has More Packets (GMP) at the end of the Scheduling Chain (SC). Thus, at each iteration of our algorithm, there will be a new GMP.

The algorithm stops when all the packets are transmitted. The effectiveness of the algorithm is proved by the value of the Standard Deviation between the TR (SDTR). Thus, in all this work, the value of SDTR is always less than 1. We expose in the next section, some previous work. In section 3, we present a CRN constituted by only one channel, one PU and several SUs. We propose a problem modeling in section 4 and expose our algorithm in section 5. In section 6, we develop some applications and finally we conclude in section 7.

2. RELATED WORKS

There are several studies in the literature which deal with this area of scheduling the users in Cognitive Radio Network like [1, 2, 3, 4, 5, 6].

In [2], the authors treated maximizing the sum of the capacity of the SUs, the stability of the queue of each SU and the equation which characterizes the transmission of the data packets. However, the authors have not given particular attention to equality between the SUs. In their approach, there is no organization of SUs relatively to the number of packets which they have to send.

In [3], the authors proposed a channel sensation to increase the transfer rate on the one hand and to use the maximum spectra on the other hand. The simulations showed the performance in terms of minimizing collisions with the PUs, maximizing throughput and resource exploitation. But the authors did not take into account the fairness between SUs in terms of services.
In [4], the authors proposed a scheduling scheme opportunistic for spectrum. In order to maximize the transfer rate, the proposed algorithm estimates the number of packets sent by each SU through each channel. But they did not address the fairness between SUs.

In [5], the authors have analyzed a cognitive radio network in which several SUs can share the same spectrum, and also one SU can use multiple spectra simultaneously in order to transmit its data. The authors seek to determine the optimal energy distribution rate for each SU in order to achieve fairness between SUs. Their approach takes into account the history of user activities to meet the QoS. To quantify the activity, they introduced the concept of dynamic weight for each SU.

In [6], the authors have used a hybrid system, to improve the performance of the SU and increase their chances of access to spectrum. They analyzed the queues of the users to show the influence of the PU on the performance of the SU.

The Performance in CRN is measured in terms of:
- Quality of Service for PU: The SU can access the spectrum only when it is not used by PU. Also, SU must leave the spectrum when a PU asks for it [6, 7]. It is necessary to avoid collision problem between users.
- Transfer rate of SUs: the SUs should send much data. In fact, The CRN that can reach a very high level is considered more effective [8, 9, 10, 11].
- Operating spectra: The exploitation of spectrum by SUs should be maximum [12, 13].
- Interference: Interference decreases the quality of service on a network. In a CRN, it is necessary to reduce interference to a minimum [14, 15].
- Energy: It is one of the most limited resources. It must be managed properly to ensure maximum service to users while consuming minimal energy [16, 17].

Our study is based on the principle of fairness between SUs. We consider that the size of the queue is an important property. For this reason, we grouped the SUs according to this property in order to transmit together their packets of data. Thus, the SUs with queues having the same size will be treated in the same way. Equity is expressed in terms of transfer rate. Our contributions include:
- Ensure the quality of services (QoS) for the PU, since the SUs operate only when the channel is free and they leave it as soon as one PU asks for it [1, 4].
- Study the problem of fairness of service between SUs, considering the combination of SU's depending on the number of packets to be transmitted, and

3. CRN WITH A SINGLE CHANNEL, A SINGLE PU AND SEVERAL SUs

Figure 1 shows a CRN with a single channel, a single PU and several SUs.

3.1 Allocation Intervals (AI)

We determine the state of the channel during the AI where AI = [kT, (k + 1)T] (Figure 2).

To specify the channel state for an AI, we introduced the notion of Indivisible Interval (II). So, we consider that the Allocation Interval (AI) can be divided into q Indivisible Intervals (II) each of which is identified by:

\[ I_h^k = \left[ kT + \frac{T}{q}h, kT + \frac{T}{q}(h+1) \right] \]  \hspace{1cm} (1)

Where \( h \in \{0, \ldots, q-1\} \)

For \( I_h^k \), the channel can take one of the two states: free or busy [2].

Let \( C \) be the channel capacity. The amount of information carried by the channel for \( I_h^k \) is given by equation (2).

\[ Q = C \times \frac{T}{q} \]  \hspace{1cm} (2)
To facilitate the study, we assume $Q$ as a measurement unit. Thus, we consider that each packet has a size $Q$. The amount of information of SU($m$) can be given by equation (3):

$$ Q_m = Q^* n_m $$

where $n_m$ is the number of packets that SU($m$) are looking to send.

For each SU, we compute the amount of information using equation (3), and then we proceed to grouping the SUs according to the number of data packets.

### 3.2 Organization of SUs

The SUs with the same number of packets are assembled in the same group (see an example in Figure 3).

![Figure 3: Grouping the SUs according to the number of packets.](image)

The table 1 below summarizes Figure 3.

**Table 1. Grouping the SUs according to the number of packets (Figure 3).**

<table>
<thead>
<tr>
<th>Group</th>
<th>Number of SUs</th>
<th>Number of Packets per SU</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G_1$</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>$G_2$</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>$G_3$</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>$G_4$</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

The groups failing to transmit all their packets during an Allocation Interval (AI) must wait the next AI. This process is repeated until all the packets will be sent.

In the proposed approach, we consider all these intervals as a single unit. We assume also that two successive packets of the same group can be sent at different AIs. The packets that are on the same line of a given group will be sent successively and without interruption.

We will focus only on packet transmission regardless of the AI used.

We model the problem in the following section. For this, we need to define some parameters.

### 4. SYSTEM MODEL

Let us note:
- $R$ is the number of groups.
- $G_i$ is a group containing $p_i$ SUs.
- $SU_{i,m}$ is a SU identified by its group $G_i$ and its index $m$.
- Every $SU_{i,m}$ has $p_i$ packets to transmit.
- $AR_i$ is the Achievement Rate of service for the group $G_i$ i.e. $AR_i$ is the number of fractions and the free time necessary to perform the service of $G_i$.
- $AR$ the vector carrying the service:

$$ AR = [AR_1, ..., AR_R]^T $$

We will formulate the problem based on the criterion of fairness of service between SUs in the following paragraph.

**4.1. Problem Formulation**

#### 4.1.1. Minimization of the difference between the transfer rates

We consider that in order to serve in the same way all SUs, we must reduce the difference between the transfer rates to the minimum possible.

Assuming that $G_i$ has $g_i$ SUs and every SU has $p_i$ packets to be transmitted. Thus, this group has $p_i g_i$ packets. If $AR_i$ is the Achievement Rate of service for the group $G_i$ and if $r_i$ is the transfer rate of the group $G_i$, then we will have:

$$ r_i = \frac{p_i g_i}{AR_i} $$

(4)

If $G_i$ and $G_j$ are two different groups, we seek to reduce the difference between the rate $r_i$ of $G_i$ and the rate $r_j$ of $G_j$ to the minimum.
The Group \( G_i \) sends all its packets after using a number of free fractions greater than \( p_i g_j \). To not monopolize the service, the group \( G_i \) must wait for other groups to send their packets. So, we can write:

\[
AR_i = p_i g_i + \sum_{j=1, j \neq i}^{R} \alpha_{ij} g_j
\]  

(6)

Let \( \alpha_{ij} g_j \) be the number of packets of \( G_j \) to send before than \( G_i \) completes its service. So, we will have \( \alpha_{ij} \leq p_j \).

If we replace in (4) \( AR_i \) by its value in (6), we obtain:

\[
r_i = \frac{p_i g_i + \sum_{j=1, j \neq i}^{R} \alpha_{ij} g_j}{p_i g_i + \sum_{j=1, j \neq i}^{R} \alpha_{ij} g_j} = \frac{1}{1 + \sum_{j=1, j \neq i}^{R} \alpha_{ij} g_j}
\]  

(7)

Note that: \( 0 < r_i \leq 1 \).

The problem can be written as follows:

\[
PB_1 = \begin{cases}
\text{Min} & r_i - r_j \\
(i, j) & \in [1, \ldots, R]^2 \\
\text{SC} & 0 \leq \alpha_{ij} \leq p_j
\end{cases}
\]

In PB\(_1\), we have \( R(R-1) \) objective functions to minimize. We seek to reformulate (PB\(_1\)) in order to reduce it to a problem with only one objective function.

**4.1.1. Reformulation of PB\(_1\)**

Let \( r^* \) be a solution of PB\(_1\), then:

\[
|r^*_i - r^*_j| \leq |r_i - r_j|, \quad \forall i, j \in \{1, \ldots, R\}
\]  

(8)

The formula (8) implies:

\[
\sum_{(i, j)}^{(R, R)} |r^*_i - r^*_j|^2 \leq \sum_{(i, j)}^{(R, R)} |r_i - r_j|^2, \quad \forall i, j \in \{1, \ldots, R\}
\]  

(9)

If we consider the following problem PB\(_2\):

\[
PB_2 = \begin{cases}
\text{Min} & \sum_{(i, j)}^{(R, R)} (r_i - r_j)^2 \\
\text{SC} & 0 \leq \alpha_{ij} \leq p_j
\end{cases}
\]

From the formula (9), it is easy to deduce that any solution of PB\(_1\) is also a solution of PB\(_2\). So to solve PB\(_1\), it suffices to solve PB\(_2\), and test the obtained solutions on the problem PB\(_1\).

The solution of PB\(_2\) is a vector with components which are very close. Therefore, they are also very close to their average.

Let \( \overline{r} \) be the average of \( r_i \), \( i \in \{1, \ldots, R\} \) and \( \sigma(r) \) the standard deviation of \( r_i \). Following [8], we can write:

\[
\overline{r} = \frac{1}{R} \sum_{i=1}^{R} r_i
\]

\[
\sigma(r) = \sqrt{\frac{1}{R} \sum_{i=1}^{R} (r_i - \overline{r})^2}
\]

(10)

(11)

The problem can be written as follow:

\[
PB_3 = \begin{cases}
\text{Min} & \sigma(r) \\
\text{SC} & 0 \leq \alpha_{ij} \leq p_j
\end{cases}
\]

The resolution of the problem PB\(_3\) must consider two conditions:

1) Optimization: to minimize the objective function.
2) Scheduling: to determine the order in which packets will be sent.

The Mathematical solving of the problem PB\(_3\) allows to determine easily the values of \( \alpha_{ij} \), for \( (i, j) \in \{1, \ldots, R\} \). But in this case, we have no information about the order in which packets will be sent. Also, if we apply the scheduling we may lose the control of the optimization.

To deal with these shortcomings, we propose in the next section an algorithm taking into account simultaneously the constraints of scheduling and the constraints of optimization. Before that, we construct a Scheduling Chain (SC).

**5. ALGORITHM**

The algorithm is a function of two arguments:

\( G \): a column vector composed of R lines containing the number of SU for each group.
P: a column vector composed of three lines containing the number of packets for SU of the same group.

This function returns the Chain as a result.

**ALGORITHM** Mono Scheduler (G, P)

**BEGIN**

1. \( GP \leftarrow [G, P] \)
2. \( GP1 \leftarrow \text{Sort}(GP) \)
3. \( [G1, P1] \leftarrow GP1 \)
4. \( \text{Chain} \leftarrow \text{empty chain} \)
5. \( St \leftarrow [G1, P1, G1*P1] \)
6. \( [u, v, w] \leftarrow St \)
7. **while** \( v \neq 0 \) **do**
   8. \( T \leftarrow \text{Max}(w) \)
   9. \( s \leftarrow \text{NumberOfLines}(T) \)
   10. **if** \( s \geq 2 \) **then**
        11. \( g \leftarrow T(o, 1) \)
        12. \( \text{Chain} \leftarrow \text{AddChain}(g, 1) \)
        13. \( St(g, 2) \leftarrow St(g, 2)-1 \)
        14. \( St(g, 3) \leftarrow St(g, 2)*St(g, 1) \)
   15. **end if**
   16. **else**
        17. \( v1 \leftarrow \text{FirstFollowingIndex}(w) \)
        18. \( g \leftarrow T(1, 1) \)
        19. **if** \( v1(1, 1) \neq 0 \) **then**
            20. \( z \leftarrow (T(1, 2)-v1(1, 3))/u(g) \)
            21. \( en \leftarrow \text{IntegerPartAdd1}(z) \)
        22. **else**
            23. \( en \leftarrow T(1, 2) \)
        24. **end if**
        25. \( \text{Chain} \leftarrow \text{AddChain}(g, en) \)
        26. \( St(g, 2) \leftarrow St(g, 2)-en \)
        27. \( St(g, 3) \leftarrow St(g, 2)*St(g, 1) \)
   28. **end if**
   29. **end while**
30. \( [u, v, w] \leftarrow St \)
31. **end while**
32. **END**

The Scheduling Chain (SC) is formed by couples. The first component of each couple is the group number and the second component is the number of packets to send. Then, after computing the number of packets of each group, we place the one which has the most packets at the end of the SC.

We also compute the number of packets that this group will keep and then we subtract this number of packets kept in order to permit to another group having more packets to transmit its packets. We repeat the procedure until all the packets will be sent.

The proposed algorithm allows establishing the SC that can be exploited to compute the rate, the inverse of transfer rate and the SDIR.

**6. APPLICATIONS**

We applied our algorithm on several instances which are randomly generated. We will show here only two instances.

**Instance 1:** Three groups and 17 samples (table 2).

The table contains four columns:
1. **1st column:** Vector G
2. **2nd column:** Vector P
3. **3rd column:** SD(P*G)
4. **4th column:** SDIR

<table>
<thead>
<tr>
<th>G</th>
<th>P</th>
<th>SD (GP)</th>
<th>SDIR</th>
</tr>
</thead>
<tbody>
<tr>
<td>[4 ;6 ;3]</td>
<td>[8;3;5]</td>
<td>7.4087036</td>
<td>0.30108</td>
</tr>
<tr>
<td>[4 ;6 ;3]</td>
<td>[10;6;12]</td>
<td>20.275658</td>
<td>0.09299</td>
</tr>
<tr>
<td>[6 ;3 ;10]</td>
<td>[9;15;22]</td>
<td>80.458409</td>
<td>0.63255</td>
</tr>
<tr>
<td>[6 ;3 ;10]</td>
<td>[5 ;6 ;10]</td>
<td>36.160138</td>
<td>0.45449</td>
</tr>
<tr>
<td>[4 ;7 ;11]</td>
<td>[12;4;16]</td>
<td>165.564218</td>
<td>0.54037</td>
</tr>
<tr>
<td>[9 ;3 ;15]</td>
<td>[5 ;8 ;13]</td>
<td>76.144599</td>
<td>0.44588</td>
</tr>
<tr>
<td>[20;13;5]</td>
<td>[15;80;31]</td>
<td>387.56361</td>
<td>0.59464</td>
</tr>
<tr>
<td>[4 ;6 ;3]</td>
<td>[8 ;4 ;6]</td>
<td>5.7348835</td>
<td>0.23545</td>
</tr>
<tr>
<td>[7 ;3 ;10]</td>
<td>[16 ;10 ;6]</td>
<td>5.7348835</td>
<td>0.46223</td>
</tr>
<tr>
<td>[4 ;6 ;10]</td>
<td>[9 ;4 ;6]</td>
<td>14.96663</td>
<td>0.28327</td>
</tr>
<tr>
<td>[3 ;7 ;12]</td>
<td>[8 ;3 ;10]</td>
<td>20.981473</td>
<td>0.46635</td>
</tr>
<tr>
<td>[4 ;8 ;12]</td>
<td>[8 ;3 ;10]</td>
<td>43.492017</td>
<td>0.45297</td>
</tr>
<tr>
<td>[6 ;4 ;12]</td>
<td>[7;13;19]</td>
<td>85.421829</td>
<td>0.64091</td>
</tr>
<tr>
<td>[2 ;4 ;8]</td>
<td>[10;14;8]</td>
<td>19.136936</td>
<td>0.16928</td>
</tr>
<tr>
<td>[4 ;8 ;12]</td>
<td>[8;14;10]</td>
<td>39.7324384</td>
<td>0.13123</td>
</tr>
<tr>
<td>[6 ;3 ;10]</td>
<td>[7;13;19]</td>
<td>70.485617</td>
<td>0.64586</td>
</tr>
<tr>
<td>[3 ;7 ;12]</td>
<td>[8;14;10]</td>
<td>41.063636</td>
<td>0.14691</td>
</tr>
</tbody>
</table>

The outcomes of interest are those of the third and fourth column. The components of SR (GP) will be sorted in descending in order to facilitate the graphical representation.

**Table 3:** Values for SD(GP) and for SDIR.

<table>
<thead>
<tr>
<th>SD(GP)</th>
<th>SDIR</th>
</tr>
</thead>
<tbody>
<tr>
<td>387.56361</td>
<td>0.59464</td>
</tr>
<tr>
<td>85.421829</td>
<td>0.64091</td>
</tr>
<tr>
<td>80.458409</td>
<td>0.63255</td>
</tr>
<tr>
<td>76.144599</td>
<td>0.44588</td>
</tr>
</tbody>
</table>
Instance 2: Five groups and 13 samples. In this case, we have a large table. It is divided into two tables: Table 4 (Containing the vector G and P) and table 5 (containing SD(GP) and SDIR).

Table 4: This table shows the values for G and P.

<table>
<thead>
<tr>
<th>G</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>[5;9;3;14]</td>
<td>[15;4;8;10;20]</td>
</tr>
<tr>
<td>[15;8;6;4;10]</td>
<td>[8;23;11;4;33]</td>
</tr>
<tr>
<td>[15;5;7;18;10]</td>
<td>[9;22;9;18;14]</td>
</tr>
<tr>
<td>[17;11;38;16;8]</td>
<td>[12;16;5;30;2]</td>
</tr>
<tr>
<td>[18;23;19;24;13]</td>
<td>[13;27;16;33;9]</td>
</tr>
<tr>
<td>[25;9;17;14;7]</td>
<td>[24;8;13;21;12]</td>
</tr>
<tr>
<td>[8;12;14;25;21]</td>
<td>[6;17;16;23;5]</td>
</tr>
<tr>
<td>[15;19;13;24;5]</td>
<td>[14;18;20;30;11]</td>
</tr>
<tr>
<td>[18;12;11;24;15]</td>
<td>[16;10;12;2;7]</td>
</tr>
<tr>
<td>[12;17;13;21;15]</td>
<td>[8;14;13;23;6]</td>
</tr>
<tr>
<td>[5;9;3;14;10]</td>
<td>[4;8;10;20;15]</td>
</tr>
<tr>
<td>[12;17;10;21;15]</td>
<td>[18;14;13;23;6]</td>
</tr>
<tr>
<td>[5;5;17;18;10]</td>
<td>[9;20;19;18;14]</td>
</tr>
</tbody>
</table>

Table 5: Values for SD(GP) and for SDIR.

<table>
<thead>
<tr>
<th>SD(GP)</th>
<th>SDIR</th>
</tr>
</thead>
<tbody>
<tr>
<td>92.7275</td>
<td>0.83</td>
</tr>
<tr>
<td>108.845</td>
<td>0.65</td>
</tr>
<tr>
<td>89.0765</td>
<td>0.82</td>
</tr>
<tr>
<td>149.7176</td>
<td>0.94</td>
</tr>
<tr>
<td>252.3652</td>
<td>0.80</td>
</tr>
<tr>
<td>192.0566</td>
<td>0.88</td>
</tr>
<tr>
<td>183.56405</td>
<td>0.78738</td>
</tr>
<tr>
<td>149.7176</td>
<td>0.9469645</td>
</tr>
<tr>
<td>144.39861</td>
<td>0.9068629</td>
</tr>
<tr>
<td>137.01036</td>
<td>0.8085614</td>
</tr>
<tr>
<td>115.9355</td>
<td>0.6487722</td>
</tr>
<tr>
<td>108.84558</td>
<td>0.6595745</td>
</tr>
<tr>
<td>96.377591</td>
<td>0.8279039</td>
</tr>
<tr>
<td>92.727558</td>
<td>0.8355148</td>
</tr>
<tr>
<td>89.076596</td>
<td>0.8275135</td>
</tr>
<tr>
<td>80.058978</td>
<td>0.7101899</td>
</tr>
</tbody>
</table>

The execution of the algorithm on different examples shows that the value of SDIR is always less than 1 (see Figure 4).

Figure 4 summarizes the previous results. The components of SD(GP) are represented on the x-axis and the components of SDIR are represented on the y-axis.

We notice that for different values of R, there is always SDIR < 1. This shows that \( \frac{1}{R} \) is too close to their average. This result holds even for large values of SD(GP).

7. CONCLUSION

In this work, we have developed a scheduling algorithm to allow transfer rates almost equal between SUs. To do this, we proposed an organization of SUs in groups. The experimental
results show that the standard deviation of the inverse transfer rate is always less than 1. This is valid even for groups with numbers of packets widely dispersed around their average. Our algorithm is general and can be adapted to any similar problem.

REFERENCES


[16] Zaheer Khan "Coordination and adaptation techniques for efficient resource utilization in cognitive radio networks" Academic dissertation presented at the Faculty of Technology of the University of Oulu, on 18 November 2011.