Assessing Performances of Incomplete DCOP Solvers on HetNet User Association Problems

Gauthier Picard[1] and Pierre Rust[2]

[1] Mines Saint-Étienne, France
picard@emse.fr
[2] Orange Labs, France
pierre.rust@orange.com

Abstract. This paper studies the applicability of distributed constrained optimization techniques (DCOP) in the context of heterogeneous radio networks (HetNets). In particular, we focus on the problem of associating user terminals to base stations, the objective of which is to determine which user (e.g. mobile phone) is associated to which base station to create a communication channel. Not all stations guarantee the same quality of service and quality of experience to users, and the quality of the final allocation (e.g. the total throughput obtained) is a criterion to be maximized. We present here a DCOP model of this association problem, which is based on an agentification of base stations coordinating to obtain an efficient allocation. We evaluate the performance of lightweight algorithms (MGM, MGM-2, DSA) on a realistic simulated environment. Their performances are compared to those of an optimal solution, as well as to those of the association policy currently used in these networks, i.e. maxSINR.

Keywords: constraint optimization · DCOP · HetNet · Resource allocation · SINR

1 Introduction

In the context of 5G communication networks, the wireless infrastructure consists of multiple base stations to which network users are associated to obtain a communication channel and thus to connect to the network and the Internet. Base stations having different characteristics (power, gain, range, etc.), we then speak of heterogeneous networks, or *HetNet*. Most target infrastructures consider multi-tier networks –e.g. 3 tiers, with long-range macro stations, medium range pico stations, and short-range femto stations, such as illustrated in Figure 1. Associating users with base stations can be seen as an allocation problem whose objectives can be to maximize the signal received by each of the user (to maximise the quality of the user's experience), or to distribute the load on the base stations so as not to reduce the throughput by station. Given the mobility of users, and therefore the need to provide solutions in a reactive manner, centralized optimization techniques are almost inapplicable in this area.

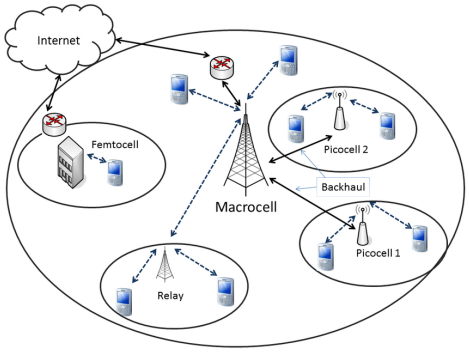

Fig. 1. Sample 3-tier HetNet architecture [4]

We propose here to evaluate the performance of distributed constraint optimization (DCOP) algorithms to this very problem, naturally distributable and modelable as a multi-agent system. Here, the base stations are coordinated via a distributed constraint optimization algorithm, to determine which users they will serve. The objective is to maximize the global throughput provided to users (i.e. the sum of all users' data rate) . The conventional approach to solve such allocation problems, and to assess quality of other approaches, is to follow a mathematical programming and combinatiorial optimization approach [11]. From an application point of view, given the hardness to solve such problems in real-time, heuristic approaches are prefered, like the greedy maxSINR approach [12] which consists in associating each terminal with the station providing the best signal over interference plus noise ratio (SINR). However, this approach has the disadvantage of associating users to top tier (e.g. macro) stations, and thus overload them, possibly causing connectivity issues. As far as we know, only [8] offers an approach to address this allocation problem, but chose a complete algorithm, BnB-ADOPT, without evaluating the performance of lighter algorithms, such as MGM or DSA. In this study, we propose to broaden our knowledge on the applicability of constraint modeling and DCOPs to this user association problem.

This article is structured as follows. The DCOP model and the algorithms used in the remainder of the paper are briefly introduced in Section 2. The HetNet user association problem, as well as the model of the system, are presented in Section 3, where constraint modeling using the DCOP formalism is also detailed. Section 4 outlines the experimental protocol and presents an analysis of the results obtained on a simulated realistic environment, according to scenarios proposed in the literature [8,2]. Finally, we conclude the article with some perspectives offered by this investigation in Section 5.

2 Distributed Constraint Optimization

This section shortly introduces the DCOP framework and some algorithms later used in the experimental part.

2.1 DCOP Model

One way to model some coordination problem between smart objects is to formalize the problem as a distributed constraint optimization problem (DCOP) [17].

Definition 1 (DCOP). *A discrete* Distributed Constraint Optimization Problem *(or DCOP) is a tuple* $\langle \mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{C}, \mu \rangle$*, where:* $\mathcal{A} = \{a_1, \ldots, a_{|A|}\}$ *is a set of agents;* $\mathcal{X} = \{x_1, \ldots, x_n\}$ *are variables owned by the agents;* $\mathcal{D} = \{\mathcal{D}_{x_1}, \ldots, \mathcal{D}_{x_n}\}$ *is a set of finite domains, such that variable* x_i *takes values in* $\mathcal{D}_{x_i} = \{v_1, \ldots, v_k\}$*;* $\mathcal{C} = \{c_1, \ldots, c_m\}$ *is a set of soft constraints, where each* c_i *defines a cost* $\in \mathbb{R}^+ \cup \{+\infty\}$ *for each combination of assignments to a subset of variables (a constraint is initially known only to the agents involved);* $\mu : \mathcal{X} \rightarrow \mathcal{A}$ *is a function mapping variables to their associated agent;* $f : \prod \mathcal{D}_{x_i} \rightarrow \mathbb{R}$ *is an objective function, representing the global cost of a complete variable assignment.*

The optimization objective is represented by function f, which, in general, is considered as the sum of costs: $f = \sum_i c_i$.

Definition 2. *A* solution *to a DCOP P is a complete assignment to all variables. A solution is* optimal *is it minimizes* f*.*

Note that the notion of cost can be replaced by the notion of utility $\in \mathbb{R}^+ \cup \{-\infty\}$. In this case, solving a DCOP is a maximization problem of the overall sum of utilities.

As highlighted by [10], DCOP have been widely studied and applied in many areas of reference. They have many interesting properties: (i) focus on decentralised approaches where agents negotiate a joint solution through local message exchanges; (ii) exploitation of the domain structure (by encoding it in constraints) to address hard computational problems; (iii) wide variety of solution methods ranging from exact methods to heuristic and approximate techniques; such as, for example, ADOPT [16], DPOP [17], MaxSum [9], DSA [23] or MGM [15], to name only the most famous.

2.2 DCOP Algorithms

There are many different algorithms for solving DCOPs. They differ by their type of resolution process (search/inference), their completeness (optimal/approachable), or their parallelism (synchronous/asynchronous) [10]. In addition, some algorithms will be able to handle certain types of constraints better than others; e.g., inference algorithms, such as MaxSum and DPOP, require exponential memory in the the arity of the constraints, therefore will be poorly adapted to problems with constraints that involve a large number of variables. We will use here two classical approximate and lightweight algorithms, namely DSA [23] and MGM [15].

DSA. More than a single algorithm, DSA (*Distributed Stochastic Algorithm*) is a family of incomplete, local search, very lightweight algorithms based a rather simple idea: agents start with a random value from their domain and regularly evaluate if the quality of their own partial assignment, defined as the total values of those constraints in which it is involved, could be improved by selecting a new value [23]. This evaluation is based on the knowledge of the values currently selected by their neighbors. If this quality can be improved, the agent decides randomly, with an *activation probability* p, to select the corresponding value and send its updated state to its neighbors.

This search process is *local* as agents base their decision only on their knowledge of the values of their direct neighbors. Of course such local search algorithm can become trapped in a local minima (even if DSA stochasticity sometime helps it escaping such local minima) and does not guarantee to find the optimal solution. Although not strictly *anytime*, DSA is an *iterative* algorithm and can be used to obtain a complete assignment at any time, in real time, with a solution quality improving, on average, over time. However, in the general case DSA provides no guarantee of monotony: as there is no coordination in the decision process and an agent's local knowledge may be outdated, two agents may simultaneously take contradictory decisions, resulting in a decrease in the overall result's quality.

Five variations –namely DSA-A, DSA-B, DSA-C, DSA-D and DSA-E– of this basic principle have been studied [23], depending on the strategy used for values change. An agent may select a new value more or less aggressively, when its state quality can be improved, strictly or not, and when where are still conflicts even if the quality cannot be improved. These variants exhibit various degree of parallelism and solution space exploration. DSA-B is considered to be the most efficient approach in the general case.

The value used for activation probability p has also been shown in [23] to have a huge influence in DSA's efficiency and quality and exhibits phase transition property. When the right variant and activation probability have been selected for a given problem class, DSA provides very good quality results, with minimal network and computational load, which makes it highly scalable.

It should be noted that DSA is able to work with n-ary constraints without any modification.

MGM. MGM (*Maximum Gain Message*) is a modification of Distributed Breakout Algorithm (DBA) that focuses on gain message passing [15,14]. As DSA, MGM algorithm is an incomplete local search algorithm that can handle n-ary constraints MGM is a synchronous algorithm: at each round, agents compute the maximum change in quality, named *gain*, they could achieve by selecting a new value. and send this gain to their neighbors. An agent is then allowed to change its value only if its gain is larger than the gain received from all its neighbors. This mechanism ensure that two variables involved in the same constraint will never change their value in the same round. This process repeats until a termination condition is met. While it provides no bounds on the solution quality, MGM

is able to guarantee monotony; eliminating the stochastic aspect of DSA ensures that the solution quality only improves over time. Monotony is a very interesting quality in many application domains, however, this quality is guaranteed at the expense of an higher tendency to become trapped in a local minima. To mitigate this issue, [15] proposes a coordinated version of MGM (usually MGM-2, but it can be extended to MGM-k), where k agents can coordinate a simultaneous (i.e. in the same round) change of values. This allows avoiding some local minima while preserving the monotony of the algorithm.

3 User Association Problem Model

This section introduces the user association problem in HetNets and details the model of the system as well as its physics. An explicit DCOP formulation is then proposed for this problem.

3.1 User Association in HetNets

The user association problem can be formulated as distributed resource allocation problem in which a unit resource refers to a resource block (RB) that covers a certain frequency range and duration [24]. To obtain a good quality of service (i.e. a high data rate), a user will have to reserve several blocks on a station, in the available limit. Indeed, the quality of service is proportional to the number of blocks reserved. The number of blocks to be reserved depends on the power of the station and the relative position of the user to the station, due to signal loss.

In recent years, significant research efforts have been made in the following areas dedicated to the development of distributed methods, in particular methods based on game theory and combinatorial optimization [3,13,11,22]. Here, the formulation of the system model is based on mixed integer linear programming [11], which is always NP-Hard. The way to get the solution is often very expensive. The simulation results in [3] and [22] show that the performance of these methods is better than some geometric and greedy methods, for example maxSINR (which assigns user to the best SINR station, see next section). However, these two approaches –game theory, and combinatorial optimization– are limited by the fact that a HetNet is a dynamic system in which the number of users and the resource configuration and allocation frequently vary for each base station.

3.2 System Model

Based on the model from [8], a tier in the HetNet indicates macrocells, picocells or femtocells, where each level contains a set of base stations having the same configurations (transmission power and resources). The resources configured at a base station refer to a set of resource blocks (RBs) where each RB consists of one of a certain duration and bandwidth [7]. Let's consider k-tier HetNet comprising base stations $\mathcal{B} = \{b_1, \ldots, b_{|\mathcal{B}|}\}$ and users $\mathcal{U} = \{u_1, \ldots, u_{|\mathcal{U}|}\}$, as shown

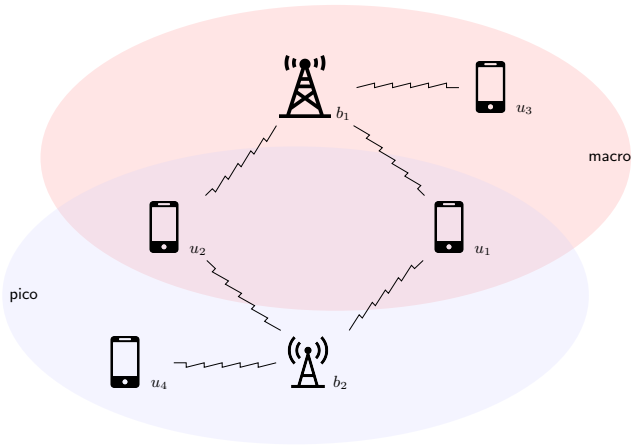

Fig. 2. Sample HetNet configuration with 4 users and 2 base stations in two different tiers

in Figure 2. Note that N_i is the number of maximum blocks on b_i. Assuming that the information on the channel are available at the base stations, the base stations on different levels also share the same total bandwidth, so that there is interference at both intra- and inter-level when base stations instantaneously allocate blocks to users. The signal-to-noise-plus-interference ratio (SINR) experienced by user u_j served by b_i in the kth tier, illustrated in Figure 3, is given by:

$$\mathrm{SINR}_{ij} = \frac{P_k g_{ij}}{\sum_{b_l \neq b_i} P_k g_{lj} + BN_0} \tag{1}$$

where P_k is the transmission power for the base stations of the kth tier, g_{ij} is the channel power gain between u_j and b_i, B is the bandwidth and N_0 is the spectral density of the noise. The channel power gain includes the effect of signal loss and its fading. We're assuming that the loss on the path is static and that its effect is reflected in the average value of the power gain of the channel, while fading is supposed to follow the exponential distribution. Second, the efficiency of user u_j associated with station b_i, noted e_{ij} is:

$$e_{ij} = \log_2(1 + \mathrm{SINR}_{ij}) \tag{2}$$

Considering bandwidth B, duration T and planning interval Γ configured for each RB, we get the unit rate at u_j on an RB as follows:

$$q_{ij} = \frac{BTe_{ij}}{\Gamma} \tag{3}$$

and thus the rate get by user u_j with n_{ij} blocks on b_j from kth tier is:

$$r_{ij} = n_{ij} q_{ij} \tag{4}$$

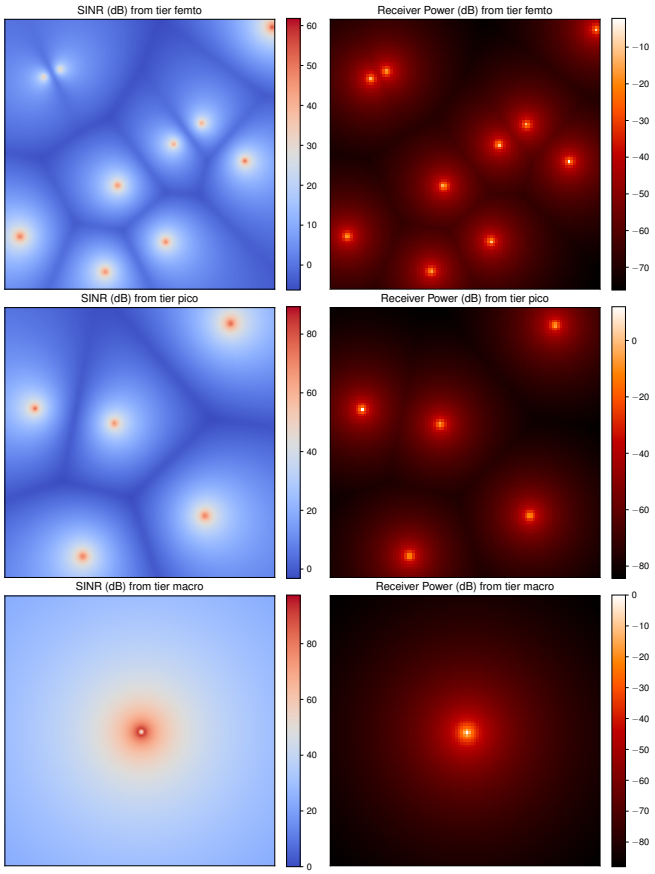

Fig. 3. SINR and received signal power at each point of the environment, for different tiers in a sample 3-tier architecture

The Quality of Service (QoS) of each user is then expressed as the minimum total rate the user will be able to obtain. In posing γ the minimum rate accepted by the users (e.g. only accept 50 Mbps services), minimum number of blocks required is estimated at:

$$n_{\min}^{ij} = \lceil \frac{\gamma}{q_{ij}} \rceil \tag{5}$$

$\lceil \cdot \rceil$ denoting the ceiling function.

Now that the physical model and the HetNet definitions have been established, we present a DCOP formulation, which specifies the different components of the DCOP, especially the constraint encoding.

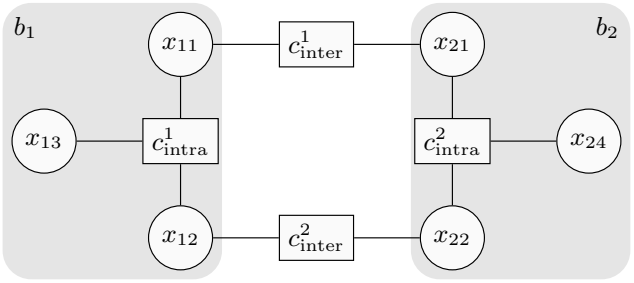

Fig. 4. Graphical representation (factor graph) of a DCOP modeling the user association problem in Figure 2

3.3 DCOP Formulation

To declare a DCOP, we have to specify the tuple $\langle \mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{C}, \mu, f \rangle$. The decision variables for a station are the set of decisions on whether or not to serve each user. Note x_{ij} the number of blocks allocated to u_j per b_i, and thus $\mathcal{X} = \{x_{ij}, b_i \in \mathcal{B}_j, u_j \in \mathcal{U}\}$, where $\mathcal{B}_j$ is the set of stations considered by u_j, and so we're looking at binary domains $\mathcal{D}_{ij} = \{0, n^{ij}_{\min}\}$. Potentially, $\mathcal{B}_j = \mathcal{B}$, but as proposed in [8], for the sake of simplicity, $\mathcal{B}_j$ can be all η best stations collected per u_j with respect to SINR, rated $\mathcal{B}^\eta_j$. DCOP constraints must ensure that the number of blocks used at each station does not exceed the maximum capacity (intra-station constraints, grouped together in $\mathcal{C}_{\text{intra}}$), and that each user is connected to at most one station (inter-station constraints, grouped together in $\mathcal{C}_{\text{inter}}$). We've declared all the DCOP constraints $\mathcal{C} = \mathcal{C}_{\text{intra}} \cup \mathcal{C}_{\text{inter}}$. With ψ_i the total number of users interested in b_j, we have $\forall c^j_{\text{intra}} \in \mathcal{C}_{\text{intra}}$:

$$R(c^j_{\text{intra}}) = \begin{cases} -\infty & \text{if } \sum_{u_j \in \psi_i} x_{ij} > R_i \\ \sum_{u_j \in \psi_i} r_{ij} & \text{otherwise} \end{cases} \tag{6}$$

and $\forall c^i_{\text{inter}} \in \mathcal{C}_{\text{inter}}$:

$$R(c^i_{\text{inter}}) = \begin{cases} 0 & \text{if } \sum_{u_j \in \psi_i} \frac{x_{ij}}{n^{ij}_{\min}} \leq 1 \\ -\infty & \text{otherwise} \end{cases} \tag{7}$$

Finally, we obtain the global objective of the DCOP:

$$\sum_{c \in \mathcal{C}} R(c) = \sum_{b_i \in \mathcal{B}} \sum_{u_j \in \mathcal{U}} r_{ij} \tag{8}$$

The objective function f is therefore the sum of the rates achieved by all users, which can be considered as a throughput measurement in HetNet. Thus, the greater the $R(c)$, the more the data rate should be good. Finally, the last determining choice is the set of agents. An intuitive approach is to consider the stations as agents who will negotiate with each other to determine which users

```yaml
name: hetnet_2_4
objective: min
agents: [macro_1, pico_1]
variables:
  v_user_0_macro_1:
    domain: d_user_0_macro_1
  v_user_0_pico_1:
    domain: d_user_0_pico_1
  v_user_1_macro_1:
    domain: d_user_1_macro_1
  v_user_1_pico_1:
    domain: d_user_1_pico_1
  v_user_2_macro_1:
    domain: d_user_2_macro_1
  v_user_2_pico_1:
    domain: d_user_2_pico_1
  v_user_3_macro_1:
    domain: d_user_3_macro_1
  v_user_3_pico_1:
    domain: d_user_3_pico_1
domains:
  d_user_0_macro_1:
    values: [0, 15]
  d_user_0_pico_1:
    values: [0, 24]
  d_user_1_macro_1:
    values: [0, 18]
  d_user_1_pico_1:
    values: [0, 31]
  d_user_2_macro_1:
    values: [0, 10]
  d_user_2_pico_1:
    values: [0, 13]
  d_user_3_macro_1:
    values: [0, 6]
  d_user_3_pico_1:
    values: [0, 7]
constraints:
  c_inter_user_0:
    function: 0 if ((v_user_0_macro_1 / 15) + (v_user_0_pico_1 / 24)) <= 1
        else 200000
    type: intention
  c_inter_user_1:
    function: 0 if ((v_user_1_macro_1 / 18) + (v_user_1_pico_1 / 31)) <= 1
        else 200000
    type: intention
  c_inter_user_2:
    function: 0 if ((v_user_2_macro_1 / 10) + (v_user_2_pico_1 / 13)) <= 1
        else 200000
    type: intention
  c_inter_user_3:
    function: 0 if ((v_user_3_macro_1 / 6) + (v_user_3_pico_1 / 7)) <= 1 else
         200000
    type: intention
  c_intra_macro_1:
    function: 200000 if (v_user_0_macro_1 + v_user_1_macro_1 +
        v_user_2_macro_1 + v_user_3_macro_1) > 200 else -((0.69 *
        v_user_0_macro_1) + (0.57 * v_user_1_macro_1) + (1.11 *
        v_user_2_macro_1) + (1.95 * v_user_3_macro_1))
    type: intention
  c_intra_pico_1:
    function: 200000 if (v_user_0_pico_1 + v_user_1_pico_1 + v_user_2_pico_1
        + v_user_3_pico_1) > 100 else -((0.43 * v_user_0_pico_1) + (0.33 *
        v_user_1_pico_1) + (0.82 * v_user_2_pico_1) + (1.65 * v_user_3_pico_1
        ))
    type: intention
```

Fig. 5. A Sample pyDCOP problem declaration for the example in Figure 2

to serve, as they are able to communicate quickly with each other, unlike users. So, here $\mathcal{A} = \mathcal{B}$. This model is illustrated in Figure 4, and declared in pyDCOP problem definition language [19] as in Figure 5. Once declared, such problem can be solved by any DCOP solution method implemented in pyDCOP.

This formulation is quite simple, however, the presence of n-ary constraints is a rather important limitation for inference-based DCOP algorithms, such as MaxSum or DPOP, that become inapplicable when n is large because they are based on the calculation of cost tables of exponential size in the arity of the constraints. Indeed, the intra-constraints will be connected to as many variables as there are users interested in a station, and in the case of a macro station, it may have to manage demands from all the users, since it will offer very good throughput. In addition, inter-constraints connect as many variables as there are stations of interest for a given user; potentially, all stations. However, this limitation of inter-station constraints can be bypassed by setting the η parameter to a limited number of candidate stations per user, e.g. 2 or 3 in [8].

3.4 About a Binary Constraint Encoding

The two techniques for binarizing constraints are (i) the addition of hidden variables for each n-ary constraint and binary constraints between each such variables or (ii) the substitution of constraints by dual variables and primal variables by binary constraints [1]. However, the newly created variables have domains that represent all the combinations of values of n variables involved in the constraint: the values of these variables are vectors of size n. Thus, the size of the domain of a dual variable representing an n-ary constraint is 2^n.

In the context of constraints on binary variables, an effective solution for storing these exponential domains is to use binary decision diagrams (or BDD) [5,8], i.e., represent all the combinations possible in the form of an acyclical oriented graph, which is a compression of the tree representing all these combinations. Thus, instead of storing a very long list of values, we store an automaton on which one can iterate to get the domain values. However, this technique requires compiling the constraints to generate the BDDs. This operation takes an exponential amount of time in the number of variables in the worst case, and the compilation times are incompatible with online operation (taking several minutes for 30 variables, whereas here we have up to ($\mathcal{U}| = 200$). Thus, we consider that a formulation with binary constraints is not suitable in our case.

4 Experimental Evaluation

Now that we have defined the HetNet user association problem, and its DCOP formulation, we evaluate the performance of various algorithms from the literature that can solve these problems in a reasonable amount of time, using the same problem declarations.

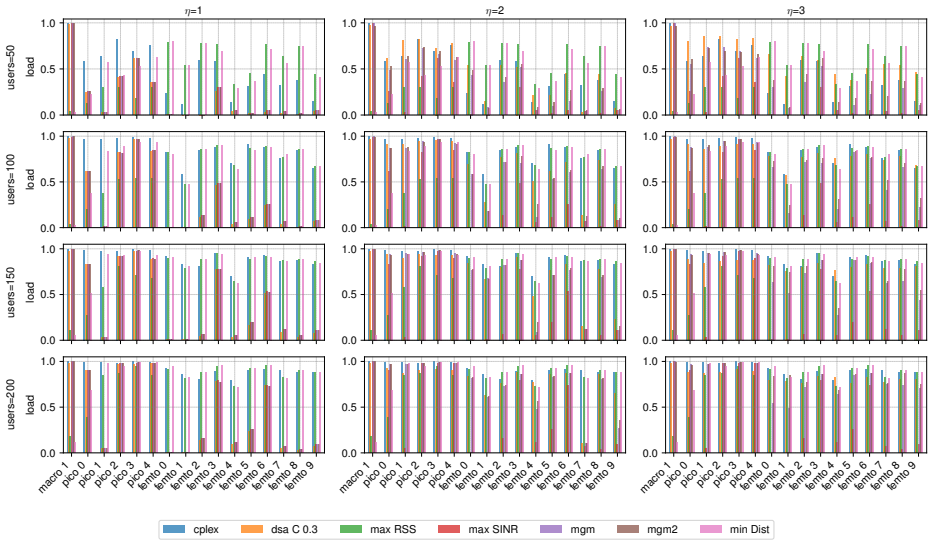

Fig. 6. Average load rate of stations obtained by each allocation algorithm for different numbers of users and different values of parameter η

4.1 Experimental Framework

We consider here the 3-tier HetNet illustrated in Figure 3, with 1 macro station, 5 pico stations, and 10 femto stations, inspired by the scenarios from the literature [8,2]. The transmission powers of these stations are 40, 35 and 24 dBm, respectively. The environment consists of a square of $1,000$ m $\times$ $1,000$ m. The macro station is located at the center, and the other stations are randomly positioned by Latin hypercube sampling. Here, we consider a number of randomly positioned users wishing to access the network $|\mathcal{U}| \in \{50, 100, 150, 200\}$. From $|\mathcal{U}| > 100$, the network is overloaded, and some users cannot be served. The signal path loss model between a macro or pico station and the users is defined as $L(d) = 34 \log_{10} 40$, with d the distance in meters between the station and the user. Similarly, for femto the path loss is $L(d) = 37 \log_{10} 30$. The noise power is -111.45dBm, which is the thermal noise at room temperature with a bandwidth of 180kHz. The time interval of the equation is $t = 1.00$ second and $\Gamma = 1.00$. The number of blocks available is 100 for macro stations, 50 for pico stations, and 25 for femto stations. We consider $\eta \in \{1, 2, 3\}$.

The following algorithms are evaluated: (i) optimal allocation, with CPLEX, by formulating the n-ary DCOP as an integer linear program; (ii) allocation using maxSINR, which amounts to associate each user with the available station having the maximum SINR (in case of of non-availability, the user switches to a second choice, etc.); (iii) allocation by maxRSS, equivalent to maxSINR, but with the power without interference or noise as a selection criterion; (iv) minDist, equivalent to maxSINR, but with the distance as selection criterion; (v) MGM;

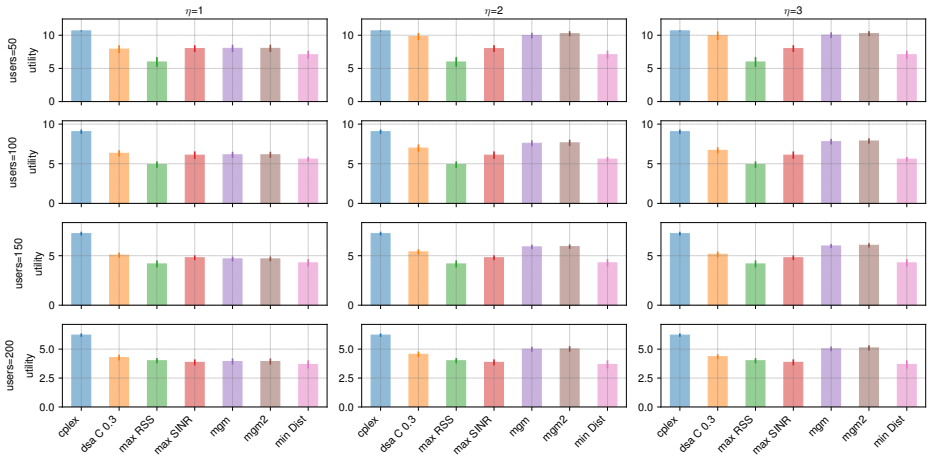

Fig. 7. Final utility (average data rate per user in Mbps) obtained by each allocation algorithm for different numbers of users and different values of parameter η

(vi) MGM-2; (vii) DSA-C (probability set at 0.3)[3]. We've narrowed it down to these algorithms, because other pyDCOP algorithms did not provide results in an acceptable time frame. (60s timeout). For every η and N, we generate 20 DCOP instances, and each instance is solved 10 times by each DCOP algorithm, starting from the allocation obtained by maxSINR. The other algorithms being deterministic, they are run only once. We display the mean and standard deviation measures on these 20 or 200 instances. MGM, MGM-2 and DSA have been parameterized to provide solutions after 30 iterations.

Calculations were performed on a computer with Intel(R) Xeon(R) CPU E5-1603 v3@2.80GHz and 32GiB of memory, using the pyDCOP library [20,19], codes written in Python 3.8, except for optimal solutions computed with CPLEX version 12.61 [6] using the Python API via the PuLP library [18].

4.2 Performance Analysis

Figure 6 shows the average load of each base station for each configuration and algorithm. As found in the literature, maxSINR implies a high load on the macro station, and then, as the number of users increases, the load is shifted to the lower tier stations. All algorithms display this behavior, except maxRSS and minDist, which mainly assign users to the closest stations, and as users are spread out over all of the environment, all stations have a significant load. For $\eta = 1$, note that DCOP algorithms have the same behavior as maxSINR, and will therefore set aside the lower tier stations, since users consider only the best station wrt SINR. With $\eta > 1$, stations start from the maxSINR allocation, then coordinate with

[3] This configuration of DSA was the most successful among the DSA family on this issue

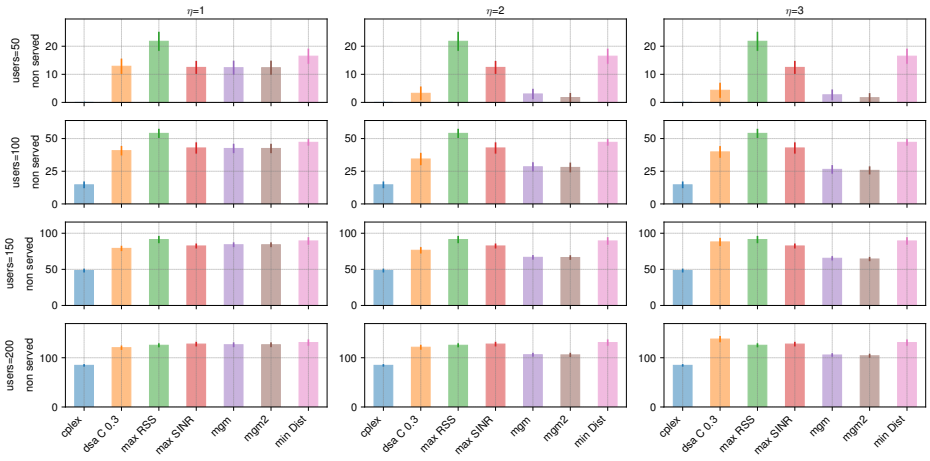

Fig. 8. Final number of non served users obtained by each allocation algorithm for different numbers of users and different values of parameter η

a DCOP algorithm to improve the solution. We then obtain allocations close to the optimal ones calculated by CPLEX, which are better from the users' point of view.

Indeed, by analyzing the overall utility of the system ($\sum_{c \in \mathcal{C}} R(c)$), i.e., the global data rate in Mbps, presented in Figure 7, we can observe that for $\eta = 1$, DCOP algorithms get the same quality as maxSINR (since there is no coordination), except for DSA. The latter, through its random moves, changes the initial maxSINR allocation, and thus removes users from some stations, and add users who were initially not served, to obtain better solutions, on average. However, this quality is not really operational, since DSA violates many intra-constraints, and thus overloads stations beyond their limits. For $\eta > 1$, the quality of MGM and MGM-2 solutions is increasing, and lies between the optimal quality and the quality of maxSINR. With increasing η, MGM-2 is providing very slightly better glabal utilities than those obtained by MGM. The worst solutions are obtained by maxRSS and minDist.

This superior quality is due to the addition of coordination between stations, which allows more users to be served, as shown in Figure 8. It shows the same performance for MGM-2 with $\eta = 3$, which is positioned between maxSINR and the theoretical optimal solution obtained by CPLEX. Note that the instances we have generated are very stressful, since not all users can be expected to be physically served, from $\mathcal{U} \geq 85$. On non-overloaded systems ($|\mathcal{U}| = 50$), MGM and MGM-2 greatly improve the proportion of served users. This gap decreases in overloaded systems, but MGM-2 still provides better solutions.

However, this coordination requires a significant network load for the stations to process coordination messages, such as illustrated in Figure 9. But note that these DCOP messages are only propagated between base stations (i.e. high

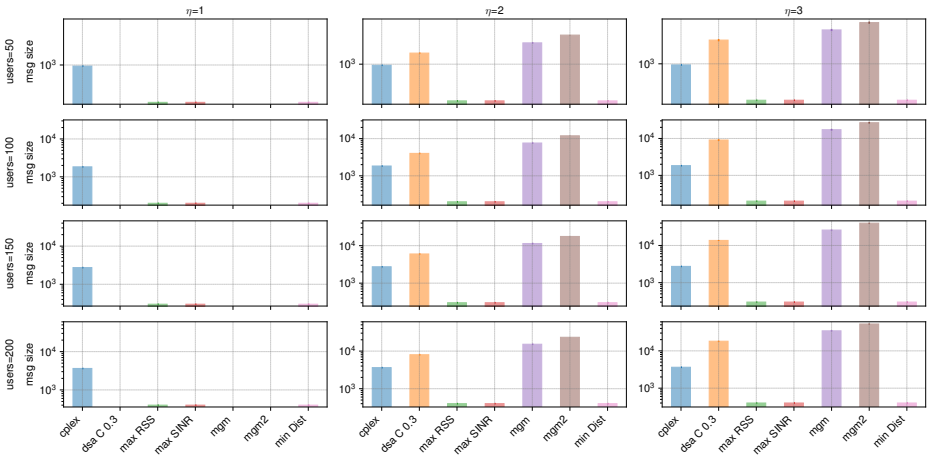

Fig. 9. Communication load induced by each allocation algorithm for different numbers of users and different values of parameter η

throughput and low latency), and especially that these protocols are low sensitive to message loss, unlike a solution that would consist of collecting all users' data (position, signal, SINR, requirements, etc.) and all base stations' data (block usage and load), then upload them to a global server to perform global optimization, with CPLEX, for example.

In the end, with MGM, for a smaller amount of information than MGM-2, good quality solutions can be obtained if message overload between stations is constraining.

5 Conclusions

In this paper, we studied the performance of lightweight DCOP algorithms for associating users in HetNets. Here, the objective is to maximise the overall throughput obtained by users. We compared MGM, MGM-2, and DSA-C algorithms to classical allocation algorithms in the field (maxSINR, maxRSS) as well as to the theoretical optimal solution obtained from centrally by mathematical programming (CPLEX). MGM and MGM-2 are well suited to perform in busy environments, placing between the classically used distributed solution, maxSINR, and the theoretical optimum. Thus, by adding a coordination layer between base stations, the system enhances the performance normally obtained in HetNets. This improvement requires message exchange only between stations (and therefore with high throughput, low latency and no bottleneck).

This study thus leads to promising results about the use of DCOP techniques in HetNet. However, we only made use of generic algorithms from the literature. In order to further improve performance, it may require algorithms and their data structures to be adapted to the particular case of HetNets; in particular, because

of the presence of n-ary constraints and the mix of soft and hard constraints. Finally, one of the envisaged contributions of a decentralized approach is the ability to adapt the system locally in the event of a failure of equipment. We plan to continue this study in the near future, using a dynamic DCOP framework, in particular the resilience mechanism proposed in [21].

References

1. F. Bacchus and P. van Beek. On the conversion between non-binary constraint satisfaction problems. In *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, AAAI '98/IAAI '98, page 311–318, USA, 1998. American Association for Artificial Intelligence.
2. B. Bikram Kumar, L. Sharma, and S.-L. Wu. Online distributed user association for heterogeneous radio access network. *Sensors*, 19(6), 2019.
3. H. Boostanimehr and V. K. Bhargava. Unified and distributed qos-driven cell association algorithms in heterogeneous networks. *IEEE Transactions on Wireless Communications*, 14(3):1650–1662, March 2015.
4. S. Boudko, W. Leister, and S. Gjessing. Heterogeneous wireless network selection: Load balancing and multicast scenario. *International Journal on Advances in Networks and Services*, 6:118–135, 12 2013.
5. R.E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, 1986.
6. `https://www.ibm.com/analytics/cplex-optimizer`.
7. A. Damnjanovic, J. Montojo, Y. Wei, T. Ji, T. Luo, M. Vajapeyam, T. Yoo, O. Song, and D. Malladi. A survey on 3gpp heterogeneous networks. *IEEE Wireless Communications*, 18(3):10–21, June 2011.
8. P. Duan, C. Zhang, G. Mao, and B. Zhang. Applying distributed constraint optimization approach to the user association problem in heterogeneous networks. *IEEE Transactions on Cybernetics*, 48(6):1696–1707, 2018.
9. A. Farinelli, A. Rogers, and N. R. Jennings. Agent-based decentralised coordination for sensor networks using the max-sum algorithm. *Autonomous Agents and Multi-Agent Systems*, 28(3):337–380, May 2014.
10. F. Fioretto, E. Pontelli, and W. Yeoh. Distributed constraint optimization problems and applications: A survey. *Journal of Artificial Intelligence Research*, 61:623–698, 2018.
11. D. Fooladivanda and C. Rosenberg. Joint resource allocation and user association for heterogeneous wireless cellular networks. *IEEE Transactions on Wireless Communications*, 12(1):248–257, 2013.
12. K. Gomadam, V. R. Cadambe, and S. A. Jafar. A distributed numerical approach to interference alignment and applications to wireless interference networks. *IEEE Transactions on Information Theory*, 57(6):3309–3322, June 2011.
13. V. N. Ha and L. B. Le. Distributed base station association and power control for heterogeneous cellular networks. *IEEE Transactions on Vehicular Technology*, 63(1):282–296, Jan 2014.
14. M. Jain, M. Taylor, M. Tambe, and M. Yokoo. DCOPs meet the real world: Exploring unknown reward matrices with applications to mobile sensor networks. In *International Joint Conference on Artificial Intelligence (IJCAI'09)*, pages 181–186, 2009.

15. R.T. Maheswaran, J.P. Pearce, and M. Tambe. Distributed algorithms for dcop: A graphical-game-based approach. In *Proceedings of the 17th International Conference on Parallel and Distributed Computing Systems (PDCS)*, pages 432–439, 2004.
16. P. J. Modi, W. Shen, M. Tambe, and M. Yokoo. ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees. *Artificial Intelligence*, 161(2):149–180, 2005.
17. A. Petcu and B. Faltings. A scalable method for multiagent constraint optimization. In *International Joint Conference on Artificial Intelligence (IJCAI'05)*, pages 266–271, 2005.
18. `https://pythonhosted.org/PuLP/`.
19. `https://github.com/Orange-OpenSource/pyDcop`.
20. P. Rust, G. Picard, and F. Ramparany. pyDCOP, a DCOP library for IoT and dynamic systems. In *International Workshop on Optimisation in Multi-Agent Systems (OptMAS@AAMAS 2019)*, 2019.
21. P. Rust, G. Picard, and F. Ramparany. Resilient distributed constraint optimization in physical multi-agent systems. In *European Conference on Artificial Intelligence (ECAI)*. IOS Press, 2020.
22. Z. Wang, X. Zhu, X. Bao, and S. Zhao. A novel resource allocation method in ultra-dense network based on noncooperation game theory. *China Communications*, 13(10):169–180, Oct 2016.
23. W. Zhang, G. Wang, Z. Xing, and L. Wittenburg. Distributed stochastic search and distributed breakout: Properties, comparison and applications to constraint optimization problems in sensor networks. *Artificial Intelligence*, 161(1-2):55–87, January 2005.
24. L. Zhao, Y. Qin, M. Ma, X. Zhong, and L. Li. Qos guaranteed resource block allocation algorithm in lte downlink. In *7th International Conference on Communications and Networking in China*, pages 425–429, Aug 2012.