E-Diophantine
Estimating Peak Allocated Capacity in Wireless Networks

Xavier Costa-Pérez[†], Zhendong Wu[†], Marco Mezzavilla[†] J. Roberto B. de Marca[⊕] and Julio Aráuz[*]

[†] NEC Laboratories Europe, Heidelberg, Germany
[⊕] Catholic University, Rio de Janeiro, Brazil
[*] School of Information and Telecommunication Systems, Ohio University, U.S.A
E-mail: {perez, wu, mezzavilla}@neclab.eu, r.demarca@comsoc.org, arauz@ohio.edu

Abstract— **Wireless networks providing QoS guarantees need to estimate the increase in peak allocated capacity when considering admitting a new resource reservation in the system. In this paper we analyze different available approaches to compute this capacity increase and, based on their limitations, propose the *E-Diophantine* solution along with two heuristics of polynomial complexity: *E-Diophantine-W* and *E-Diophantine-UW*. The properties of the designed algorithms are derived through a mathematical analysis and their accuracy and computational load characteristics evaluated in a generic scenario. Complementary to the generic study, a network performance evaluation comparing the different approaches is conducted using OPNET's simulator and considering a realistic wireless network.**

Based on our results, the main conclusions that can be drawn are: i) the larger the degree of flexibility allowed for defining the resource reservations characteristics, the larger the potential benefit of the *E-Diophantine* solutions both in accuracy and computational load terms and ii) for systems supporting a large number of reservations, the *E-Diophantine* heuristics can be used to reduce the computational load from exponential to polynomial (cubic) at a low estimation error probability cost.

Keywords—**LTE, Wireless MAN, Wireless LAN, 3GPP, WiMAX, IEEE 802.16, IEEE 802.11, QoS, Admission Control.**

I. Introduction

Wireless networks are a key element of today's society to communicate, access and share information. The ever increasing wide range of services and applications building on wireless network capabilities results in a diverse range of Quality of Service (QoS) requirements that needs to be fulfilled to ensure user satisfaction. Third (3G) and Fourth-generation (4G) broadband wireless technologies such as 3GPP Long Term Evolution (LTE) [1], LTE-Advanced [2], IEEE 802.16 [3] and IEEE 802.16m [4] have already defined flexible mechanisms to be able to support a large number of different QoS requirements. In the Wireless Local Area Network (WLAN) domain, a similar path was followed with the standardization of IEEE 802.11e [5] and IEEE 802.11-2012 [6]. Complying with the QoS requirements of granted service demands is mandatory for service providers and requires accurately estimating the available system capacity when deciding whether new service requests can be accepted. Precise capacity estimation techniques allow for the design of efficient admission control algorithms in order to maximize the utilization of networks while ensuring a satisfactory Quality of Experience (QoE) for users.

In this paper we propose a peak allocated capacity estimation algorithm, *E-Diophantine*, and evaluate its performance. This new algorithm improves the accuracy of prediction versus complexity trade off when compared with currently available approaches. The results here presented extend our previous work in [7] by: i) analyzing the *E-Diophantine* complexity as compared to competing approaches and ii) designing and evaluating two heuristics, *E-diophantine-W* and *E-diophantine-UW*, that reduce the *E-Diophantine* computational load from exponential to polynomial (cubic) at a low estimation error probability cost.

This article is structured as follows. In Section II we review the standardized QoS resource reservation specifications for wireless networks, propose a common modeling and describe solutions available in the literature to estimate peak allocated capacity. Our proposed *E-Diophantine* approaches are explained in Sections III and IV along with their mathematical foundations and corresponding complexity analysis. The allocated capacity estimation accuracy and computational load of the different solutions is compared and a realistic wireless system evaluation performed in Section V. Finally, Section VI contains a summary of our findings and concludes the paper.

II. Estimation of Peak Allocated Capacity

Wireless networks support QoS reservation of resources by allowing new flows to apply for admittance in the system through request messages indicating their specific requirements. Such requests contain a set of QoS parameters which include different information depending on the service type. In the following we review the different QoS reservation schemes as defined by the predominant wireless technologies in order to find out their commonalities. We take here an industry-driven top-down approach where, rather than considering the information we would ideally like to have for our modeling, we analyze which information is actually available based on the standardized specifications of the wireless technologies under consideration.

A. Reservation Information Available based on Standardized Specifications

3GPP LTE

We start our analysis considering 3GPP's LTE, the wireless cellular technology becoming predominant worldwide. QoS reservations in LTE's evolved packet system (EPS) are based on bearers which correspond to packet flows established between the packet data network gateway (PDN-GW) and the mobile stations. The bearer management and control follows the network-initiated QoS control paradigm and defines two types of bearers: Guaranteed bit rate (GBR) and Non-Guaranteed bit rate (Non-GBR). Bearers are assigned a scalar value referred to as a QoS class identifier (QCI). Several standardized QCI values with specific characteristics have been defined to allow for successful multivendor deployment and roaming. Table I summarizes these standardized QCIs as described in [8].

3GPP LTE Standardized QCI Parameters					
Res. Type	**QCI**	**Min. Rate**	**Delay**	**Loss Rate**	**Service Example**
GBR	1	•	100 ms	10^{-2}	Conv. Voice
GBR	2	•	150 ms	10^{-3}	Live Streaming
GBR	3	•	50 ms	10^{-3}	Real-time Gaming
GBR	4	•	300 ms	10^{-6}	Buffered Streaming
Non-GBR	5		100 ms	10^{-6}	IMS Signaling
Non-GBR	6		300 ms	10^{-6}	TCP-based apps

TABLE I
3GPP LTE STANDARDIZED QCI PARAMETERS

WiMAX

Second, we consider WiMAX networks as the major LTE competing technology. WiMAX supports QoS reservation of resources by allowing a new flow to apply for admittance in the system through a Dynamic Service Addition REQuest message (DSA-REQ). Such requests contain a QoS parameter set which includes different mandatory information depending on the data delivery service requested in the downlink direction (DL), Base Station (BS) to Subscriber Station (SS), or the scheduling service requested in the uplink direction (UL). Five different QoS services are supported: Unsolicited Grant Service (UGS), Extended Real-Time Variable Rate Service (ERT-VR), Real-Time Variable Rate Service (RT-VR) and Best Effort Service (BE). Table II summarizes the required QoS parameter set per Data Delivery Service according to the IEEE 802.16 standard [3]. A similar set of parameters is required in the uplink direction.

IEEE 802.16 QoS Parameters per Data Delivery Service					
Data Delivery Services	**UGS**	**ERT-VR**	**RT-VR**	**NRT-VR**	**BE**
Min. Resv. Tr. Rate (MRTR)	•	•	•	•	
Max. Sust. Tr. Rate (MSTR)		•	•	•	•
SDU size	•				
Maximum Latency	•	•	•		
Tolerated Jitter	•	•			
Traffic Priority		•		•	•
Req./Trans. Policy	•	•	•	•	•

TABLE II
WIMAX QOS PARAMETERS PER DATA DELIVERY SERVICE

Wireless LAN

Finally, we consider the Wireless LAN technology as the most popular wireless technology in homes and hotspots nowadays. In Wireless LAN QoS reservation of resources has been also enabled as defined by IEEE 802.11e [5] and 802.11-2012 [6]. These standards introduce the Traffic Specification (TSPEC) mechanism which defines a set of mandatory QoS parameters in admissible TSPECs for different service types. In Table III we summarize a subset of these admissible TSPECs.

B. Model for Peak Allocated Capacity Estimation

Based on the aforementioned set of parameterized QoS guarantees for 3GPP LTE, WiMAX and Wireless LAN, we identify a minimum common subset of QoS parameters available based on the standardized specifications composed by i) starting time of granted QoS reservation request, ii) required data rate and iii) periodicity at which the guaranteed data rate should be fulfilled (delay bound). Based on this minimum common subset of information available in the QoS reservations, in the following we define a reservation model which will be used for estimating the peak of allocated capacity when considering admitting a new reservation.

IEEE 802.11-2012 Subset of Parameters in Admissible Traffic Specifications				
TSPEC	**MSDU Size**	**Mean Rate**	**Delay Bound**	**Burst Size**
Continuous	•	•	•	
Controlled	•	•	•	
Bursty				•
Contention	•	•		

TABLE III
IEEE 802.11-2012 SUBSET OF PARAMETERS IN TRAFFIC SPECIFICATIONS

Let us consider an arbitrary reservation r_i for which a minimum set of requirements can be defined for services with QoS guarantees as:

- Reservation start time: t_i (ms)
- Reservation required capacity: B_i (bits)
- Periodicity of reserved resources: T_i (ms)

Then, based on these requirements, our QoS reservation model can be described as follows: given a reservation starting time t_i, a certain amount of capacity B_i (bits) is reserved periodically for transmitting reservation r_i data within a time interval T_i. Based on this reservation model, a resource reservation request r_i can be expressed as a periodic discrete sequence of Kronecker deltas with amplitude B_i in the following way

$$r_i(t) = B_i \cdot \delta_{t,t_i+n\cdot T_i} = \begin{cases} B_i & \text{if } t = t_i + n \cdot T_i \;;\; n \in \mathbb{Z}_{\geq 0} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Once a new reservation request is received, an allocated capacity estimation algorithm needs to evaluate the impact of accepting it on the currently existing aggregated allocated capacity peak. Assuming a wireless system with N reservations already granted, we define $A(t)$ as the aggregation (as a function of time) of the N flows already in the system plus the one requesting admittance. See Figure 1 for a graphical representation of the model for a 5 reservations example.

The objective of our estimation algorithm is thus, starting from a given point of time, t_0, find the allocated capacity peak, $max(A(t))$, when considering the new reservation request. This can be expressed as an optimization problem as follows:

$$\begin{aligned} \max_{t} \quad & A(t) = \sum_{i=1}^{N+1} B_i \cdot \delta_{t,t_i+n\cdot T_i} \\ \text{s.t} \quad & t \geq t_0 \end{aligned}$$

In the next section we will consider different possible approaches available in the literature to estimate the peak allocated capacity at any given point of time, $max(A(t))$, according to the reservation model defined in Eq.1 and their dependence with the *granularity* defined. We define *granularity* (Gr) as the minimum unit that can be used for setting t_i and T_i. This parameter is necessary to account for the time-slotted scheme used in the

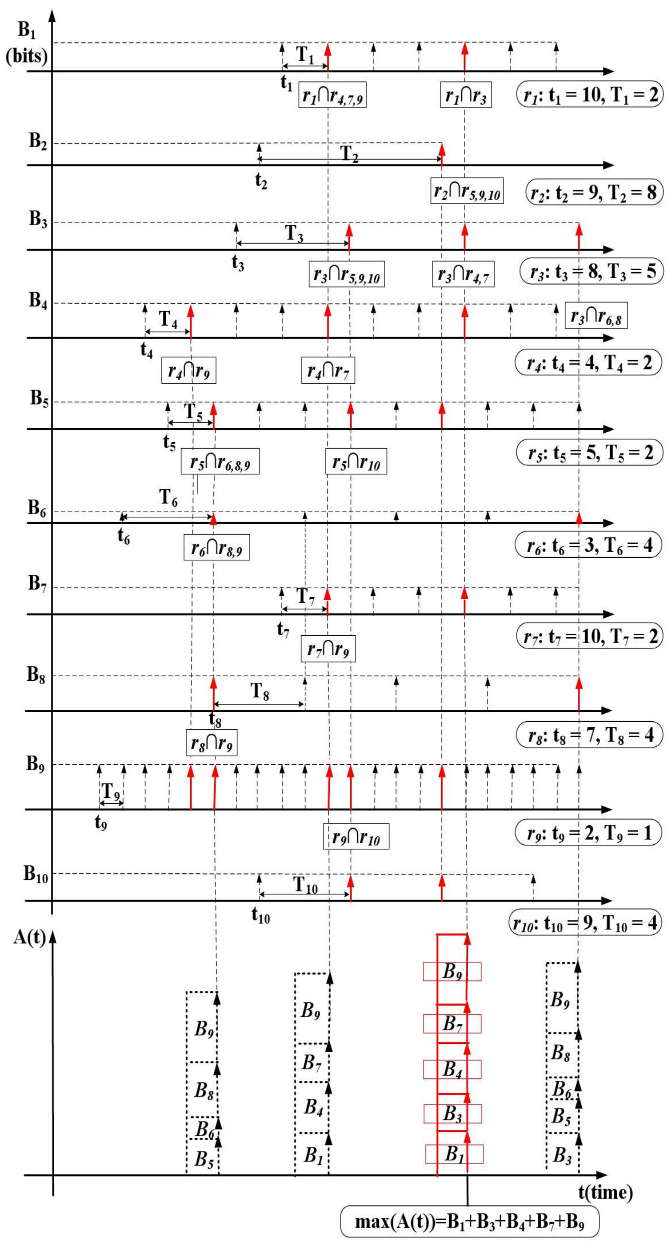


Fig. 1. Illustration of QoS reservation model for a 5 reservations example considering granularity 1 and identification of reservations needs overlap $(r_i \cap r_j)$ in time.

real wireless systems under consideration which define admitted frame/slot durations.

C. Related Work

An extensive body of literature exists in the area of reserved capacity estimation solutions for admission control purposes in wireless networks. In the following we review some of the main approaches related to the objectives of our work which focuses on providing deterministic QoS guarantees based on explicitly signaled reservation information. Note that, given the large amount of related works, this summary is not meant to be exhaustive but representative of the available solutions. Well-known literature not specifically designed for providing deterministic QoS guarantees as *Effective Bandwidth based* approaches, e.g., [9], [10] or *Measured Throughput based* solutions, e.g., [11], have been omitted for the sake of brevity.

C.1 Throughput-based

Throughput-based approaches, e.g., [12], consider the mean or peak data rate requirements specified by an application in order to estimate the peak allocated capacity.

In order to determine $max(A(t))$ the *Throughput-based* approach assumes that all admitted reservations need to be served simultaneously, i.e., without taking into account the time at which flows actually need to be served. The following equation corresponds to the *Throughput-based* approximation of $A(t)$.

$$A_{Thr} = \sum_{i=1}^{N+1} B_i \tag{2}$$

Such an approach requires few computational resources; however, it is neither well suited to take into consideration the bandwidth-varying nature of typical applications such as video, nor the times at which resources are required. Thus, in peak-based implementations, the actual available resources are likely to end up underutilized while, in mean-based implementations, QoS guarantees might be jeopardized.

Throughput-based Solution Complexity

The complexity of this solution increases linearly with the number of reservations considered and therefore, can be expressed as $O(N)$.

C.2 Throughput and Periodicity-based

Throughput and Periodicity-based approaches, e.g., [13] and [14], take into account not only the rate requirements of the reservations but also the delay constraints and/or periodicity of requests obtained from a traffic description included in the admittance request. These solutions make use of the knowledge of the least common multiple (LCM) of the periods of the different accepted reservations and consider it for estimating the peak allocated capacity.

This *Throughput and Periodicity-based* approach obtains an accurate solution for $max(A(t))$ by computing all values of $A(t)$ within a T_{LCM} period. Note that since $A(t)$ is composed of *N+1* periodic reservations, its period T_{LCM} corresponds to the Least Common Multiple (LCM) of the periods of the reservations in the system plus the one under consideration. The following optimization problem formulation corresponds to the $max(A(t))$ computation in this case.

$$\begin{aligned} \max_{t} \quad & A(t) = \sum_{i=1}^{N+1} B_i \cdot \delta_{t, t_i + n \cdot T_i} \\ \text{s.t} \quad & t \geq t_0 \\ & t \leq T_{LCM} \end{aligned}$$

This approach, although accurate, has a dependence with the LCM of the reservations in the system which, depending on the granularity allowed Gr, might increase exponentially with the number of reservations and thus, become too expensive in computational terms. Therefore, such a solution might not be feasible in practice unless a limitation in the granularity of periods is imposed. This issue will be studied in Section V-A.

Throughput and Periodicity-based Solution Complexity

The complexity of the *Throughput and Periodicity-based* algorithm increases according to the number of reservations $N+1$ and their corresponding LCM as $(N+1) \cdot LCM$. In the worst case, the LCM increases with the number of flows as $\frac{1}{Gr}\prod_{i=1}^{N+1} T_i$ resulting in a complexity of $O(\frac{N+1}{Gr}\prod_{i=1}^{N} T_i) \rightarrow O((T_{max})^N)$, where T_{max} corresponds to the maximum period of the reservations under consideration.

C.3 Diophantine

In order to remove the LCM dependency of the *Throughput and Periodicity-based* approach, another solution is considered based on *Diophantine* theory which, in general, deals with indeterminate polynomial equations and allows variables to be integers only. In the rest of the paper this approach will be referred as *Diophantine* and it has already been considered in [15] as a solution for estimating the peak allocated capacity in wireless networks.

We define this solution as follows. Considering a flow already accepted in the system described with the resource reservation $r_i(t) = B_i \cdot \delta_{t,t_i+n_i \cdot T_i}$ and a new flow requesting admittance characterized by $r_j(t) = B_j \cdot \delta_{t,t_j+n_j \cdot T_j}$, the peak $B_i + B_j$, will occur for the set of n_i and n_j combinations which fulfill

$$\{n_i, n_j \in \mathbb{Z}_{\geq 0} : t_i + n_i \cdot T_i = t_j + n_j \cdot T_j\} \tag{3}$$

Based on the previous result, we define the term *diophantine set* as that composed of non-negative integer solutions for n_i and n_j satisfying the condition in Eq.3. In order to find this set of solutions Eq.3 can be expressed as the following linear diophantine equation with two variables

$$\{n_i, n_j \in \mathbb{Z}_{\geq 0} : n_i \cdot T_i - n_j \cdot T_j = t_j - t_i\} \tag{4}$$

Then, based on linear diophantine equations theory, we know that there will be a set of integer solutions for n_i and n_j if

$$\frac{t_j - t_i}{d} \in \mathbb{Z} \tag{5}$$

where $d = gcd(T_i, T_j)$ and gcd stands for greatest common divisor.

When the previous condition holds, the *diophantine set* of solutions corresponding to a specific pair of reservations can be found with the extended Euclidean algorithm which will find a and b such that

$$a \cdot T_i + b \cdot T_j = d \; ; \text{ where } a,\ b \ \in \ \mathbb{Z} \tag{6}$$

The diophantine set corresponding to the intersection of reservations i and j, indicated with the subindex ij, can be then expressed as follows

$$\{n_{ij} \in \mathbb{Z}_{\geq 0} : t_{ij} + n_{ij} \cdot T_{ij}\} \;\; ; \text{ where } T_{ij} = lcm(T_i, T_j) \tag{7}$$

where the smallest n_i and n_j $\in \mathbb{Z}_{\geq 0}$ satisfy

$$t_i + n_i^{min} \cdot T_i = t_j + n_j^{min} \cdot T_j \equiv t_{ij} \tag{8}$$

By applying the $Diophantine$ solution to all pairs of reservations in the system, as well as to their found *diophantine sets* in a recursive manner, an exact solution for $max(A(t))$ can be found which is independent of the LCM length.

Diophantine Solution Complexity

This solution requires to compute the gcd for all pairs of reservations in the system as well as the sets of intersections, *diophantine sets*, found. Therefore, the total number of gcd executions in the worst case corresponds to

$$\sum_{i=1}^{N+1} \binom{N+1}{i} \tag{9}$$

Eq. 9 presents a maximum at $i = \lceil \frac{N+1}{2} \rceil$ because of its symmetric distribution and, for a fixed value of $N+1$, it can be expressed as 2^{N+1}. Thus, since the number of operations to compute the gcd is upper bounded by $log_2(T_{max})$, where T_{max} corresponds to the maximum period of the pairs of reservations or intersections under consideration, the complexity of the *Diophantine* solution is $O(2^{N+1} \cdot log_2(T_{max})) \rightarrow O(2^N)$.

Note that since the computational complexity increases exponentially as the number of reservations grows, the Diophantine solution might become also unfeasible in practice.

III. E-Diophantine

Based on the performance issues identified for the approaches described in the previous section in [7] we proposed an enhancement to the *Diophantine* solution, hereinafter referred as *E-Diophantine*. The objective is to achieve the same accuracy when estimating the aggregated allocated peak capacity but at a lower computational cost.

The *E-Diophantine* solution proposed consists in first, exactly as in the original case, finding the sets of intersections, *diophantine sets*, between all pair of reservations under consideration. After this step, instead of repeating the process in a recursive manner for all diophantine sets found, the results are structured in what we will refer to in the rest of the paper as *matrix of intersections* of reservations.

We define the *matrix of intersections* as a matrix where the information regarding the intersections found between all pairs of reservations in the system are indicated with a 1 when there is an intersection and 0 otherwise. See Table IV for a 10 reservations example where the values for t_i and T_i of each reservation were randomly drawn from a uniform distribution between 1 and 10. Based on this matrix, the rest of the sets are derived based on the information obtained regarding the reservations involved in each intersection.

In [7] we provided the theorems and proofs that enable the described *E-Diophantine* solution. For the sake of brevity this informations is not repeated here. The interested reader is referred to [7] for further details on the analytical demonstration.

	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}
r_1	1	0	1	1	0	0	1	0	1	0
r_2	0	1	1	0	1	0	0	0	1	1
r_3	1	1	1	1	1	1	1	1	1	1
r_4	1	0	1	1	0	0	1	0	1	0
r_5	0	1	1	0	1	1	0	1	1	1
r_6	0	0	1	0	1	1	0	0	0	0
r_7	1	0	1	1	0	0	1	0	1	0
r_8	0	0	1	0	1	0	0	1	1	0
r_9	1	1	1	1	1	0	1	1	1	0
r_{10}	0	1	1	0	1	0	0	0	0	1

TABLE IV

EXAMPLE OF MATRIX OF INTERSECTIONS OF RESERVATIONS

A. E-Diophantine Algorithm

Algorithm 1 details the steps followed by the *E-Diophantine* solution. The first part of the algorithm, which finds the diophantine sets, is identical to the *Diophantine* approach. Once the sets have been obtained, a matrix of intersections is constructed by simply structuring the diophantine sets results regading the intersections between each pair of reservations in a matrix form. This operation corresponds to the function $compute\_matrix\_inters(.)$ in the algorithm and an example is shown in Table IV for a set of 10 reservations.

The problem of finding the aggregated peak allocated capacity is similar to the *clique problem* which consists in finding particular complete subgraphs ($cliques$) in a graph, i.e., sets of elements where each pair of elements is connected [16], [17]. The clique problem is based on an adjacency matrix where the edges between vertices are given for a specific graph indicating their interconnection. This adjacency matrix can be mapped in our case to the matrix of intersections where edges represent intersections between reservations. Our goal is then finding a complete subgraph containing the *weighted maximum clique*. This problem has been extensively researched in the past and several solutions are available in the literature, see for instance [18].

The *E-Diophantine* algorithm designed is based on the general backtracking solution [19]. It consists in a recursive algorithm which incrementally builds, on a reservation by reservation basis, a set of candidate intersections keeping for each reservation the aggregated peak resource reservation in a vector called $solutions$. The solutions obtained starting the backtracking process at each different reservation are kept in $solutions$ and once all have been explored the function $find\_maximum(solutions)$ simply selects the maximum value contained in this matrix.

In order to improve the efficiency of the $backtracking\_search(.)$ function, we introduce the condition $weight+solution(next) \leq max$ in Algorithm 1 where $weight$ corresponds to the aggregate bandwidth in the current candidate set while max corresponds to the aggregated allocated capacity peak found so far. Based on this condition we can determine if the consideration of a new reservation in our current candidate set will result in a larger aggregated resource reservation peak. If the condition is true we can skip backtracking based on this reservation and move on to the next one. The same logic applies to the condition $weight + max\_potential(start) \leq max$ where we evaluate at each iteration whether it is still possible for the current set

Algorithm 1 **E-Diophantine** algorithm to find out the peak resource requirement for a new reservation r_{N+1} with starting time t_{N+1}, period T_{N+1} and requirement B_{N+1} considering the set of N reservations already accepted in the system with their corresponding starting times $t = (t_1 ... t_N)$, periods $T = (T_1 ... T_N)$ and requirements $B = (B_1 ... B_N)$

Call executed for each new reservation request
for $i = 1$ to $N+1$ **do**
  for $j = i+1$ to $N+1$ **do**
    if $solution\_exists(t_i, t_j, T_i, T_j)$ **then**
      $intersections \leftarrow find\_inters\_dioph(t_i, t_j, T_i, T_j)$
    end if
  end for
end for
$m\_inters \leftarrow compute\_matrix\_inters(intersections)$
$m\_inters \leftarrow order(m\_inters)$
for $root = N+1$ to 1 **do**
  $solutions(root) \leftarrow backtracking\_search(root, B_{root}, max)$
end for
return $find\_maximum(solutions)$

Function backtracking_search(start,weight,max)
$next \leftarrow find\_next\_inters(start, m\_inters)$
if $next == NULL$ **then**
  if $weight > max$ **then**
    $max = weight$
  end if
end if
while $next \neq NULL$ **do**
  if $weight + solutions(next) \leq max$ **then**
    return
  end if
  if $weight + max\_potential(start) \leq max$ **then**
    return
  end if
  $backtracking\_search(next, weight + B_{next}, max)$
  $next \leftarrow find\_next\_inters(next, m\_inters)$
end while
end function

considered to be above the peak found in previous iterations.

Finally, before starting the backtracking algorithm based on the matrix of reservations, we order the reservations based on their potential aggregated peak resource reservation $B_{pot}(resv)$. The operation corresponds to the function $order(.)$ and its advantage is that it increases the probability of finding the peak solution during the first backtracking operations. Thus, reducing the need of backtracking iterations to complete the full exploration of possible solutions.

Figure 2 illustrates the tree of solutions found based on the matrix of intersections in Table IV and applying the *E-Diophantine* algorithm. As an example, in the case of reservation 1, r_1, there is no branch of solutions with r_2 since according to the matrix of intersections r_1 does not intersect with r_2. In the case of the branch of solutions $r_1 \rightarrow r_3 \rightarrow r_4$ the continuation with the branch of r_5 is also discarded since reservations r_1 and r_3 do not intersect with r_5.

E-Diophantine Solution Complexity

The first part of the *E-Diophantine* algorithm, which finds the first set of intersections (diophantine sets) between all pairs of reservations, requires in the worst case the same number of operations as the first part of the *Diophantine* algorithm: $\binom{N+1}{2} \cdot log_2(T_{max})$.

After this, both algorithms differ. In the the *E-Diophantine* case, the set of intersections found are used to construct a matrix of intersections. Based on this matrix, the rest of the inter-

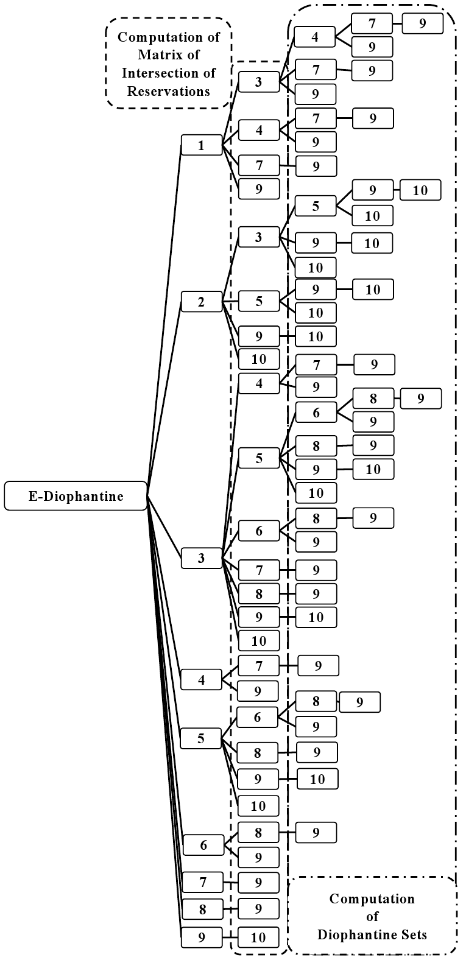


Fig. 2. Illustration of tree of solutions of intersections for the E-Diophantine algorithm based on the matrix of intersections shown in Table IV

sections are derived by exploring the tree of possible solutions, $solutions\_tree$ in Algorithm 1, without requiring the *gcd* computation.

As aforementioned, the exploration of the rest of possible intersections in order to find the peak resource requirement is similar to solving the *weighted maximum clique* problem for an arbitrary graph. In our particular case, this problem can be translated to the *unweighted maximum clique* problem (all weights are equal) by defining a reference resource requirement unit B_{ref} and then mapping the resource reservations requests in the matrix of intersections according to $\lceil B/B_{ref} \rceil$. Thus, resulting in an increase of the number of reservations of:

$$N+1 \rightarrow (N+1)^* = \sum_{i=1}^{N+1} \left\lceil \frac{B_i}{B_{ref}} \right\rceil \tag{10}$$

$(N+1)^*$ can be then expressed as $(N+1)^* = k*(N+1)$, where k is a constant controlled by the network operator.

Efficient exact algorithms for the *unweighted maximum clique* problem have been extensively researched in the literature. [20] represents one of the fastest solutions known today which incurs in a complexity of $O(2^{\frac{N}{4}})$. Considering the complexity of both parts of the *E-Diophantine* algorithm together, the overall complexity can be expressed as $O(2^{\frac{k \cdot N}{4}})$.

The worst case *E-Diophantine* complexity thus can be potentially worse than the *Diophantine* one depending on the k value determined by the operator configuration. However, in contrast to the generic *maximum clique* case, in our particular problem we do not deal with *random* edges between vertices and this can be exploited to obtain better performance. These differences will be explained in more detail in the next section and their actual performance compared in Section V-A.

IV. E-Diophantine Heuristics: E-Diophantine-W and E-Diophantine-UW

In the previous section we have shown that the *E-Diophantine* exact solution might still become unfeasible in practice due to its exponential growth with the number of reservations. In the following we describe two non-exact heuristics designed to overcome this limitation by performing a *partial* exploration of the tree of solutions instead of an *exhaustive* one. The proposed approaches achieve a polynomial computational load increase (cubic) based on the number of reservations at a bounded probability estimation error cost.

Note that, as explained in Section III-A, the *E-Diophantine* exact solution can be mapped to the unweigthed maximum clique problem which is NP-hard and therefore, no polynomial solutions are expected to be found. The difference of this problem to our particular case is that we can exploit the fact that, in our cliques, we do not deal with *random* edges between vertices but with edges determined by common properties based on our matrix of intersections construction.

The *E-Diophantine* heuristics are based on the following observation: Let us assume that three reservations r_i, r_j and r_k result in two diophantine sets $\{t_{ij} + n_{ij} \cdot T_{ij}\}$ and $\{t_{ik} + n_{ik} \cdot T_{ik}\}$ having reservation r_i in common. In such a case, since both diophantine sets are a contained inside the set $\{t_i + n_i \cdot T_i\}$ it can be intuitively observed that the probability of reservations r_j and r_k of intersecting with each other is higher than if they would not have a reservation in common. Moreover, the larger the number of reservations in common between diophantine sets, the larger the probability of intersecting with each other. The actual probability though can not be computed unless specific assumptions are made on the way the starting times and periods of reservations are chosen.

A. E-Diophantine Weighted (E-diophantine-W)

As aforementioned, a reduction of the computational load of *E-diophantine* can be achieved by performing a partial exploration of the tree of solutions instead of an exhaustive one. The approach taken in order to select which branches to explore and which ones to discard is the following and is described in pseudo-code in Algorithm 2.

First, the matrix of intersections is built in the same way as described for the *E-Diophantine* algorithm. Then, for each reservation we start the process of selecting which branch from the possible solutions tree is evaluated next until no more reservations can be considered. The selection process consists in first creating, for each reservation, a *candidate list* with the rest of the reservations with which it could potentially intersect and

Algorithm 2 **E-Diophantine-W** algorithm to find out the peak resource requirement for a new reservation r_{N+1} with starting time t_{N+1}, period T_{N+1} and requirement B_{N+1} considering the set of N reservations already accepted in the system with their corresponding starting times $t = (t_1...t_N)$, periods $T = (T_1...T_N)$ and requirements $B = (B_1...B_N)$

Call executed for each new reservation request
for $i = 1$ to $N + 1$ **do**
  for $j = i + 1$ to $N + 1$ **do**
    if $solution\_exists(t_i, t_j, T_i, T_j)$ **then**
      $intersections \leftarrow find\_inters\_dioph(t_i, t_j, T_i, T_j)$
    end if
  end for
end for
$m\_inters = compute\_matrix\_inters(intersections)$
for $i = 1$ to $N + 1$ **do**
  $initiate(black\_list, candidate\_lists)$
  while $cnt + size(black\_list) < N + 1$ **do**
    $candidate\_lists \leftarrow find\_next\_candidate(m\_inters, i)$
    $update(black\_list, candidate\_lists)$
    $cnt = cnt + 1$
  end while
end for
return $find\_maximum(candidate\_lists, B)$

a *black list* with the reservations which should be discarded. This operation corresponds to the function $initiate(.)$. After this, for each reservation in the candidate list the function $find\_next\_candidate(.)$ computes its *Weight* considering the potential aggregated peak allocated capacity that would be required by this reservation and removing the reservations present in the *black list*. Once the *Weight* for all reservations has been computed, the reservation with the largest *Weight* is selected as the next branch to be explored and the *black list* updated accordingly by the function $update(.)$. The process is then iterated until no more reservations can be considered.

B. E-Diophantine Unweighted (E-diophantine-UW)

A complementary solution to *E-Diophantine-W* can be designed by converting our *Weighted Maximum Clique* problem to an *Unweighted* one by applying a transformation to the matrix of intersections as indicated in Eq. 10. Compared to the *Weighted* problem the advantage of this transformation is that our algorithm can concentrate in finding the branch of the tree of solutions with the largest length without having to consider the actual *Weight* of them. In the downside though a larger number of reservations might need to be considered depending on the k value determined by the network operator B_{ref} configuration.

We apply this principle to design an alternative to *E-diophantine-W* which we will refer to in the rest of the paper as *E-Diophantine Unweighted (E-diophantine-UW)*. In this case the same Algorithm 2 can be used by transforming the matrix of intersections used as input according to Eq. 10 and simplifying the $find\_next\_candidate(.)$ function such that it neglects the *Weight* information.

C. E-Diophantine Heuristics Solutions Complexity

The first part of the *E-Diophantine-W* and *E-Diophantine-UW* algorithms is identical to the *Diophantine* one and requires $\binom{N+1}{2} \cdot log_2(T_{max})$ operations to obtain the matrix of intersections.

In addition to this operation, Algorithm 2 explores for each reservation $(N + 1)$ the rest of potential intersecting reservations (N). Then, for each of them a comparison is performed considering the candidate and black lists (N) in the function $find\_next\_candidate(.)$. Thus, the worst case complexity of the optimization algorithms can be approximated as $O(N^3)$ for *E-Diophantine-W* and $O((k \cdot N)^3)$ for *E-Diophantine-UW*.

D. E-Diophantine Heuristics: Probability of Estimation Error

Since the *E-Diophantine* heuristics described in the previous section do not perform an exhaustive search through the matrix of intersections but a partial one, it could happen that these solutions underestimate the actual aggregated allocated capacity peak. In this section we analyze the probability of this case to happen. We focus in *E-Diophantine-UW* for analysis simplicity reasons. A similar analysis can be applied to the *E-Diophantine-W* case.

In order to calculate the probability of *E-Diophantine-UW* selecting a branch of solutions ($selected$) different than the one containing the actual peak (max), we consider a matrix of intersections of $N \times N$ reservations and two possible branches containing different sets of intersecting reservations $S_{selected} = I_{s_1}, ...I_{s_N}$ and $S_{max} = I_{m_1}, ...I_{m_N}$; where I takes the corresponding binary value indicating whether an intersection between two flows occur based on the matrix of intersections. Then, we define the probability of a reservation of intersecting with another as P_x and i and j as the total number of intersections in $S_{selected}$ and S_{max}, respectively. Thus, assuming that the intersections between reservations are independent from each other[1], the probability of the S_{max} branch containing the maximum number of intersections, k, instead of $S_{selected}$ can be expressed as follows

$$P_{\epsilon_{sel}} = (P_x)^j \cdot (P_x)^{2(k-1)} \cdot (P_x)^i \sum_{l=i-k+1}^{i-2} (1 - P_x)^{2l} \quad (11)$$

where $i \geq j \geq k$, $P_x < 1$, $i < N$ and $k < N$

Since our objective is to find an upper bound for the error probability, Equation 11 can be simplified by assuming $j = k$ because the lower the value of j, the higher the value of $P_{\epsilon_{sel}}$

$$P_{\epsilon_{sel}} = (P_x)^{3k+i-2} \sum_{l=i-k+1}^{i-2} (1 - P_x)^{2l} \quad (12)$$

where $i \geq k$, $j = k$, $P_x < 1$, $i < N$ and $k < N$

Equation 12 provides the error probability of selecting a branch from the tree of solutions that does not contain the maximum. Since an error when selecting the next branch could occur in k to i branch selection operations in the worst case, the overall error probability can be expressed as

$$P_{\epsilon} = \sum_{s=k}^{i} (P_x)^{3k+s-2} \sum_{l=s-k+1}^{s-2} (1 - P_x)^{2l} \quad (13)$$

[1] Note that in general the instant at which a set of users in a system start using services (t_i, t_j, ...) and the periodicity in the need of resources of each service (T_i, T_j, ...) is uncorrelated and thus, the intersection between their reservations as well.

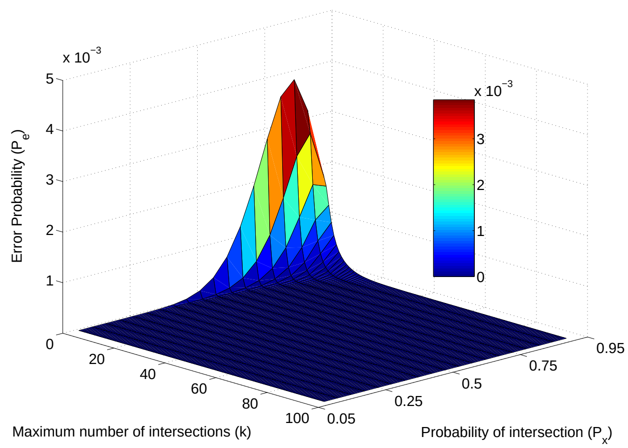


Fig. 3. Illustration of the error probability according to Eq.13 for $j = i = k$ and a wide range of P_x and k values.

where $i \geq k$, $j = k$, $P_x < 1$, $i < N$ and $k < N$

Based on Eq.13 we plot in Figure 3 for illustration purposes the value of P_ϵ for a wide range of maximum number of intersections (k) and probabilities of intersections (P_x) considering $j = k = i$. Note that this case represents an upper bound since the larger the values of i and j above k, the lower the value of P_ϵ. Moreover, in order to compute the actual probability of different reservations of intersecting with each other, specific assumptions regarding the way their starting times and periods would be required. Thus, a sweep on different possible values has been performed to cover most cases.

As it can be observed from the figure, the *E-Diophantine* heuristics introduce a low error probability ($P_\epsilon < 0.1\%$) and present the interesting property that, as the maximum number of intersections increases (relevant case for admission control decisions), P_ϵ decreases, independently of the probability of intersection value.

V. Performance Evaluation & Discussion

A. Estimation Algorithms of Peak Allocated Capacity

In the previous sections we have described different options to compute the peak allocated capacity based on QoS reservation information as defined by the currently most deployed wireless standards. In this section we compare the performance of these algorithms both in estimation accuracy and computational load terms to provide an insight on the trade-offs to be considered when deciding which option is more appropriate for a specific network deployment.

In order to evaluate the performance differences between the *Throughput-based*, *Throughput and Periodicity-based*, *Diophantine* and *E-Diophantine* approaches, we implemented these algorithms in $Matlab$ and designed the following experiment. We considered a system with 10 to 100 reservations where, for each one, t_i and T_i were randomly drawn from a uniform distribution ranging from the $granularity$ value chosen up to 100 (in multiples of the granularity). Three different granularity values were evaluated: 1, 5 and 10. B_i was randomly drawn from a uniform distribution ranging from 1 to 100. Figure 4 summarizes the results of the experiment after running a minimum of 100 seeds for each value. Note that the confidence intervals are not displayed in this case because their superimposition with the symbols of the different approaches would significantly degrade the readability of the Figures.

In Figure 4(a) the difference between the estimated maximum number of resources required by each of the approaches can be observed. Taking the *Diophantine* results as reference since it represents the exact solution, as expected, the *Throughput-based* approach is the one presenting the largest difference to the actual values; reaching differences of more than 100% in some of the cases. Such a large estimation deviation to the actual value would obviously result in available resources being underutilized and thus, in a lower revenue for a network operator.

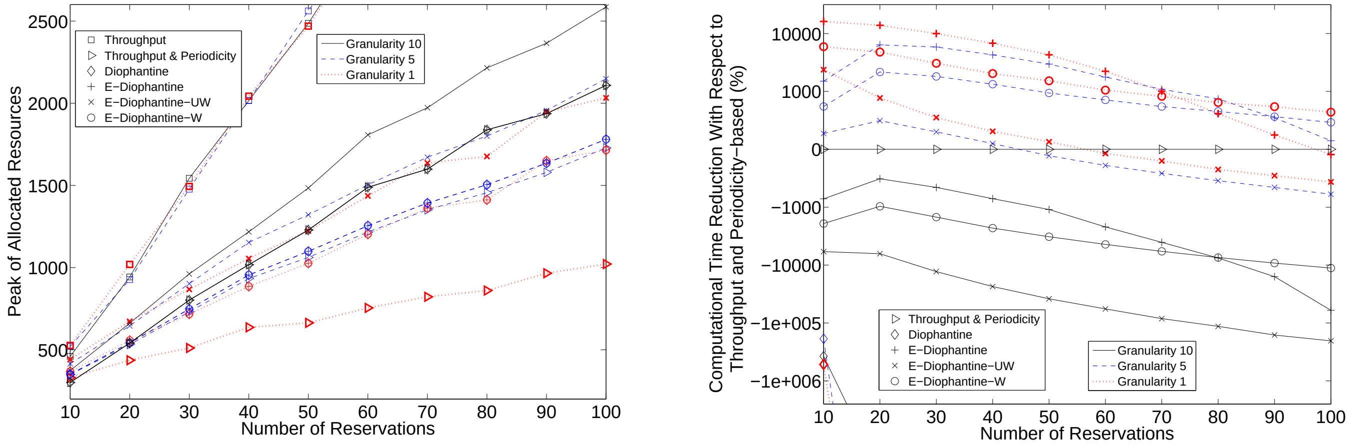
In the *Throughput and Periodicity-based* case, the smaller the granularity, the larger the difference to the correct value due to a limitation in the maximum T_{LCM} value that can be considered in a real implementation (10^7 in our system). Furthermore, the estimation is below the actual value and therefore, its usage for admission control purposes could compromise the QoS guarantees. Regarding the *E-Diophantine* estimation, as expected, it is always equal to the exact value (*Diophantine*). Finally, the *E-Diophantine* heuristics present a different accuracy depending on the solution considered. While *E-Diophantine-W* closely matches the correct prediction, *E-Diophantine-UW* overestimates because of the B_{ref} chosen (25 in this experiment). This B_{ref} value was chosen as an intermediate value to illustrate the trade-off to be considered when configuring the *E-Diophantine-UW* algorithm. The larger the B_{ref} value, the smaller the computational load increase as compared to the *E-Diophantine-W* solution but at the cost of accuracy loss.

Regarding the computational complexity, Table V summarizes the worst case complexity of the different solutions considered as derived in Sections II and III. Since the upper bounds provided in Table V correspond to the worst case performance, in this experiment we analyze the differences to be expected when considering a wide range of possibilities regarding the number of flows in the system. In Figure 4(b) the computational load differences are shown, computed as the percentage of reduction achieved with respect to the *Throughput and Periodicity-based* approach, taken here as reference due to its implementation simplicity[2].

As it can be observed, the *Diophantine* solution, although exact, exceeds by far the computational load of the alternative solutions considered and thus, it might not be feasible in practice [3]. For the largest granularity considered (*granularity 10*), the *Throughput and Periodicity-based* approach clearly outperforms in computational time the *E-Diophantine* solutions with no loss of accuracy. However, as the granularity considered decreases, the *E-Diophantine* solutions reduction with respect to the *Throughput and Periodicity-based* one increases. In particular, the *E-Diophantine* algorithm presents the largest computational load advantage as compared to the *Throughput*

[2]The *Throughput-based* case is not considered since its computational load is low but its estimation accuracy very poor (see Fig. 4(a)).

[3]For instance, in the 30 reservations case with granularity 10, the computation time in a $2 \cdot$ Quad Core simulation server took $>$1000 seconds.



(a) Peak of resources allocated for services with QoS guarantees

(b) Relative computational time of the different estimation algorithms

Fig. 4. Comparison of maximum allocated capacity estimation accuracy and computational load of Throughput-based, Throughput- and Periodicity-based, Diophantine and E-Diophantine Approaches

	Complexity
Throughput-based	$O(N)$
Throughput and Periodicity-based	$O((T_{max})^N)$
Diophantine	$O(2^N)$
E-Diophantine	$O(2^{\frac{k \cdot N}{4}})$
E-Diophantine-W	$O(N^3)$
E-Diophantine-UW	$O((k \cdot N)^3)$

TABLE V
WORST CASE COMPLEXITY OF THE DIFFERENT SOLUTIONS CONSIDERED

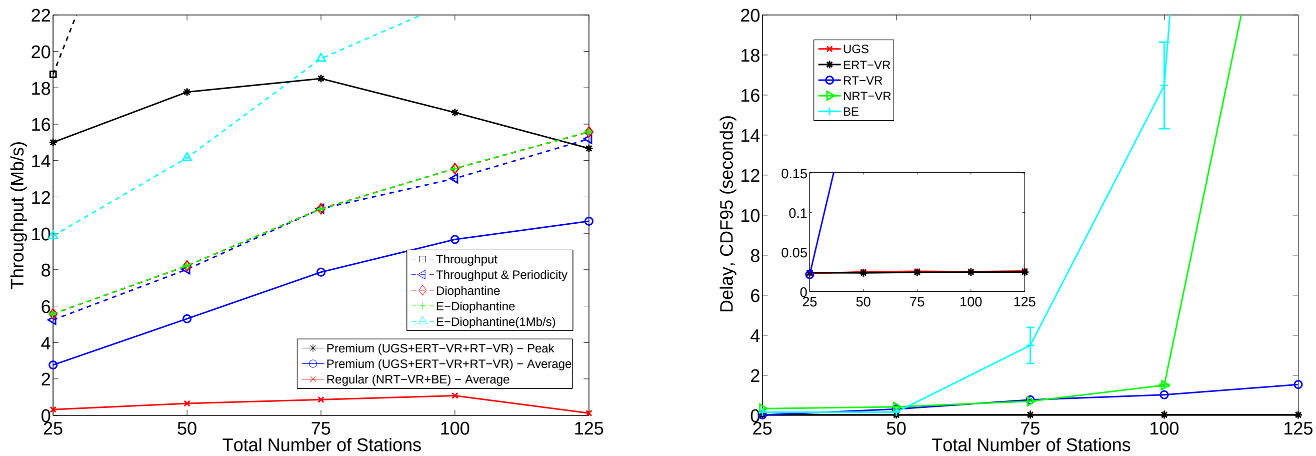
and Periodicity-based but this advantage eventually starts decreasing as the number of reservations grows. With respect to the *E-Diophantine* heuristics, while *E-Diophantine-W* manages to keep a very considerable advantage even for a large number of reservations (100), the *E-Diophantine-UW* advantage starts lower and eventually gets outperformed by the *Throughput and Periodicity-based* algorithm. The reason is the increase in the number of reservations to be considered by the the *E-Diophantine-UW* approach which grows according to Eq. 10.

B. *Wireless Network Performance Evaluation*

In the previous section we have analyzed the performance of the proposed *E-Diophantine* solution as compared to its alternatives considering a generic scenario. In this section, we complete this evaluation by considering additional elements in the performance comparison that could have an impact in the allocated peak capacity estimation of the different approaches. Examples of these elements are: wireless physical channel, Transport layer, Network layer, MAC layer, control plane signaling, realistic applications, QoS scheduler, number of stations, etc. In order to achieve this, from the different wireless technologies mentioned in Section II supporting services with QoS guarantees, WiMAX is selected as a representative example and OPNET's simulator [21] as evaluation tool for analyzing an IEEE 802.16 system. The *E-Diophantine* solution used is *E-Diophantine-W* because of its better scalability properties.

An 802.16 simulation scenario is setup according to Table VI consisting of one Base Station (BS) and multiple Subscriber Stations (SS) where each station is configured to send and receive traffic from their corresponding pair in the wired domain of its type of application, i.e., one station sends and receives Voice traffic (without silence suppression), a second station sends and receives Voice traffic (with silence suppression), a third one receives a Video stream, a fourth one does an FTP download and the last one does Web browsing. The number of subscriber stations is increased in multiples of five stations up to 125 in total, always keeping the relation of $1/5$ of stations of each application type. The QoS scheduling policy chosen is Strict Priority applied first to fulfill the Minimum Reserved Traffic Rates (MRTRs) and then, the Maximum Sustained Traffic Rates (MSTRs). The length of the simulations performed is 120 seconds with a warm-up phase of 10 seconds. In the case of the delay performance metric, the values represent the 95th percentile of the delay (CDF95) considering all simulation runs. Regarding the confidence intervals, they are shown only in the cases where their superimposition with the symbols of the different approaches does not degrade their readability. The configuration used for the different applications is detailed below:

- *Voice* G.711 Voice codec. Data rate: 64kb/s. Frame length: 20ms. Mapped to the *UGS* service in the downlink (BS $\rightarrow$ SS) and uplink direction.
- *VAD (Voice with Activity Detection)* G.711 Voice codec. Data rate: 64kb/s. Frame length: 20ms. Talk spurt exponential with mean 0.35 seconds. Silence spurt exponential with mean 0.65 seconds. Mapped to *ERT-VR* in the downlink and to *ertPS* in the uplink.
- *Video* MPEG-4 real traces [22].Target rate: 450 kb/s. Peak: 4.6 Mb/s. Frame generation interval: 33ms. Mapped to *RT-VR* in the downlink and to *rtPS* in the uplink.
- *FTP* Download of a 20MB file. Mapped to *NRT-VR* in the downlink and to *nrtPS* in the uplink.
- *Web Browsing* Page interarrival time exponentially distributed with mean 60s. Page size 10KB plus 20 to 80 objects of a size uniformly distributed between 5KB and 10KB [23]. Mapped to the BE service both in the downlink and uplink direction.



(a) Downlink Throughput

(b) Downlink Delay

Fig. 5. IEEE 802.16 Scenario: Network Performance

IEEE 802.16 Configuration					
Base Freq. (GHz)		2.5	DL Subfr. # Subch.		30
Bandwidth (MHz)		10	UL Subfr. # Subch.		35
Frame Duration (ms)		5	# Data Subc./Subch		24
Symbol Duration (μs)		102.86	# SSs 64 QAM (3/4)		60%
Number of Subcarriers		1024	# SSs 16 QAM (3/4)		30%
DL Subfr. # Symbols		35	# SSs QPSK (1/2)		10%
UL Subfr. # Symbols		12	# SSs per Scenario		25-125
Multipath Channel		ITU Ped-B	Pathloss Model		Erceg
Data Delivery Services			**Scheduling Services**		
UGS	MRTR	80 Kb/s	UGS	MRTR	80 Kb/s
	Max. Lat	20 ms		Max. Lat	20 ms
ERT-VR	MRTR	80 Kb/s	ertPS	MRTR	80 Kb/s
	Max. Lat	20 ms		Max. Lat	20 ms
RT-VR	MSTR	2 Mb/s	rtPS	MSTR	800 b/s
	MRTR	500 Kb/s		Max. Lat	33 ms
	MRTR	800 b/s		Max. Lat	20 ms
E-Diophantine Configuration					
UGS	B_{UGS}	1600 bits	UGS	B_{UGS}	1600 bits
	T_{UGS}	20 ms		T_{UGS}	20 ms
ERT-VR	B_{ERT}	1600 bits	ertPS	B_{ERT}	1600 bits
	T_{ERT}	20 ms		T_{ERT}	20 ms
RT-VR	B_{RT}	16500 bits	rtPS	B_{RT}	800 bits
	T_{RT}	33 ms		T_{RT}	1 s

TABLE VI
PERFORMANCE EVALUATION PARAMETERS

IEEE 802.16: System Performance Results

The objective of the results presented in this section is to corroborate that the E-diophantine solution successfully predicts the peak allocated capacity according to the configured MRTRs. Once the agreed aggregated MRTR capacity can not be served anymore for all classes, a sharp increase in the delay of the different traffic classes is expected to be observed according to their actual priority.

In Figure 5(a) we show the peak and average throughput experienced in the downlink by the different application types as compared to the peak capacity estimations of the different approaches described in Sections II and III. The throughput of the differents applications is aggregated according to whether it is *Premium* traffic (UGS+ERT-VR+RT-VR) or *Regular* traffic (NRT-VR+BE).

From the performance results in Fig.5(a) the first remarkable result is that the peak of *Premium* traffic is in some cases above the peak estimated with the algorithms considered but the *Throughput-based* one. The reason for this result is the 2Mb/s MSTR configured for RT-VR which allows video applications to get more than its 500 Kb/s MRTR if there is leftover capacity after serving all MRTRs. Note that the *Throughput-based* estimation is too conservative and therefore, it will not be considered in the remainder of this section. Also note that the difference between the *Throughput and Periodicity-based* and *Diophantine* result is caused by the LCM upper limit configured for the former, set to half of the actual required one in order to illustrate its effect on the estimation accuracy.

As the number of stations increases, the difference between the allocated peak capacity estimations and the throughput peak of *Premium* traffic decreases. Note that the larger the amount of *Premium* traffic in the network, the lower the opportunities to go above the MRTR value. Eventually a point is reached where even the MRTR guarantees can not be satisfied, see crossing point between 100 and 125 stations. Moreover, as the number of flows in the system increases, the signaling overhead required for the DL-MAP increases as well, resulting in a lower *Premium* average throughput. For illustration purposes, an additional *E-diophantine* case has been added, *E-Diophantine (1Mb/s)*, where the MRTR for RT-VR has been configured to 1 Mb/s instead of 500 Kb/s. This case provides an example of how the allocated capacity peak estimation would vary by allowing bursty traffic to transmit significantly above their average.

With respect to the delay performance, the results are shown in Fig. 5(b). As expected, when the wireless resources become scarce, the delay experience degrades according to the traffic priority. In the case of RT-VR traffic, in contrast to UGS and ERT-VR, the delay experienced increases constantly. This is due to the performance metric chosen, 95% percentile of the delay (CDF95), which yields a close to worst case delay for each application traffic and thus, as the number of flows grows, it increasingly represents the Video peaks that can not be absorbed because there is not enough remaining capacity after serving all MRTRs. The delay performance of BE, which increases very rapidly, is due to the simple QoS scheduling policy used, Strict

Priority, resulting in BE traffic being served only if the rest of the available traffic has already been served. Finally, the NRT-VR delay performances experience an extreme degradation after the 100 stations point. Note that this is where the peak estimation of the different algorithms but the *Throughput-based* crosses the *Premium* peak throughput and therefore, the probability of NRT-VR traffic to be served decreases significantly.

The performance results corresponding to the uplink direction have been omitted due to space restrictions since in this case there is always enough capacity to satisfy the needs of all application flows.

VI. Summary & Conclusions

Wireless networks providing QoS guarantees require an algorithm to estimate the increase in aggregated peak allocated capacity if a new resource reservation is admitted in the system. In this paper we have proposed the *E-Diophantine* solution, along with its mathematical foundations, and evaluated its benefits as compared to alternative approaches of increasing complexity, namely: *Throughput-based*, *Throughput and Periodicity-based* and *Diophantine*. The performance comparison comprised both accuracy and computational load analysis in a generic scenario as well as an evaluation using OPNET's simulator in a realistic wireless network.

The main conclusions that can be drawn from our results are: i) *E-Diophantine* solutions can be successfully used to predict the allocated peak capacity demand of admitted QoS reservations in wireless networks, ii) the *Throughput and Periodicity-based* approach can outperform *E-Diophantine* in computational terms if limitations in the period between resource allocations can be imposed, iii) the larger the degree of flexibility allowed for defining the resource reservation periods, the larger the potential benefit of the *E-Diophantine* solutions both in accuracy and computational load terms and iv) for systems supporting a large number of reservations, the *E-Diophantine* heuristics can be used to reduce the computational load from exponential to polynomial (cubic) at a low estimation error probability cost.

References

[1] Third Generation Partnership Project (3GPP), "Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Terrestrial Radio Access Network (E-UTRAN). Overall Description Stage 2," 3GPP TS 36.300 v 9.10.0, January 2013.

[2] Third Generation Partnership Project (3GPP), "Requirements for further advancements for Evolved Universal Terrestrial Radio Access (E-UTRA, lte-advanced," 3GPP TR 36.913 v 11.0.0, September 2012.

[3] IEEE 802.16 Working Group, "IEEE Standard for Local and Metropolitan Area Networks. Part 16: Air Interface for Broadband Wireless Access Systems," IEEE Standard 802.16-2009, May 2009.

[4] IEEE 802.16 Working Group, "IEEE Standard for Local and Metropolitan Area Networks. Part 16: Air Interface for Broadband Wireless Access Systems, Amendment 3: Advanced Air Interface," IEEE P802.16m-2011, May 2011.

[5] IEEE 802.11 Working Group, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications. Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements," IEEE Standard 802.11e, November 2005.

[6] IEEE 802.11 Working Group, "IEEE Standard for Local and metropolitan area networks. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications.," IEEE Std 802.11-2012, March 2012.

[7] X. Pérez-Costa, M. Mezzavilla, J.R.B. de Marca, and J. Aráuz, "E-Diophantine: An Admission Control Algorithm for WiMAX Networks," in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, Sydney, Australia, April 2010.

[8] Third Generation Partnership Project (3GPP), "Technical Specification Group Services and System Aspects. Policy and charging control architecture," 3GPP TS 23.203 v.9.13.0, September 2013.

[9] A.I. Elwalid and D. Mitra, "Effective Bandwidth of General Markovian Traffic Sources and Admission Control of High Speed Networks," *IEEE/ACM Transactions on Networking*, vol. 1, no. 3, 2002.

[10] J.S. Evans and D.Everitt, "Effective Bandwidth Based Admission Control for Multi-Service CDMA Cellular Networks," *IEEE Transactions on Vehicular Technology*, vol. 48, no. 1, 2002.

[11] S. Jamin, S.J. Shenker, and P.B. Danzig, "Comparison of Measurement-Based Admission Control algorithms for Controlled-Load Service," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, Kobe, Japan, April 1997.

[12] H. Wang, W. Li, and D.P. Agrawal, "Dynamic Admission Control and QoS for 802.16 Wireless MAN," in *Proceeding of Wireless Telecommunications Symposium (WTS)*, Pomona, USA, April 2005.

[13] C.Ciconetti, L.Lenzini, E.Mingozzi, and G.Stea, "Design and Performance Analysis of the Real-Time HCCA Scheduler for IEEE 802.11e WLANs," Elsevier Computer Networks Journal (CN), Volume 51, Issue 9, June 2007.

[14] S. Chandra and A. Sahoo, "An efficient call admission control for IEEE 802.16 networks," in *In Proceedings of the 15th IEEE Workshop on Local and Metropolitan Area Networks*, Princeton, USA, June 2007.

[15] O.Yang and J.Lu, "Call Admission Control and Scheduling Schemes with QoS support for Real-time Video Applications in IEEE 802.16 Networks," In IEEE Journal of Multimedia, May 2006.

[16] R. D. Luce and A. D. Perry, "A Method of Matrix Analysis of Group Structure," In Psychometrika Journal, June 1949.

[17] F. Harary and I. C. Ross, "A Procedure for Clique Detection Using the Group Matrix," In Sociometry Journal, 1957.

[18] D.-Z. Du and P. M. Pardalos, "Handbook of Combinatorial Optimization - The Maximum Clique Problem," Kluwer Academic Publishers, 1999.

[19] C.Stein T.H. Cormen; C.E. Leiserson, R.R. Rivest, "Introduction to Algorithms," The MIT Press, 2001.

[20] J. M. Robson, "Finding a maximum independent set in time $O(2^{n/4})$," Tech. Rep., http://www.labri.fr/perso/robson/mis/techrep.html, 2001.

[21] "OPNET Simulator," http://www.opnet.com.

[22] F.H.P. Fitzek and M.Reisslein, "MPEG-4 and H.263 Video Traces for Network Performance Evaluation," IEEE Network,Vol. 15,No. 6,pages 40-54, November/December 2001.

[23] "Website Optimization," http://www.websiteoptimization.com/speed/tweak/.