Faster Deterministic Wakeup in Multiple Access Channels*

Gianluca De Marco †‡ Marco Pellegrini † Giovanni Sburlati †

Abstract

We consider the fundamental problem of waking up n processors sharing a multiple access channel. We assume the weakest model of synchronization, the locally synchronous model, in which no global clock is available: processors have local clocks ticking at the same rate, but each clock starts counting the rounds in the round in which the correspondent processor wakes up. Moreover, the number n of processors is not known to the processors. We propose a new deterministic algorithm for this problem, which improves on the currently best upper bound.

keywords: algorithms, clock, synchrony, wakeup problem, multiple access channel.

1 Introduction

We consider the following model of multiple access channel, taken as the basis for theoretical studies on radio networks, satellite channels, serial bus communication networks. There are n processors (stations) sharing a common communication channel. The communication system is synchronous, in the sense that the processors send messages in rounds. Once a message is written on the channel, the message is broadcast to all other processors. Unfortunately, since the channel is shared by all processors, a collision among several processors attempting to write in the same round might occur. Precisely, the channel has the following property: a message is written successfully on the channel in a given round (and therefore heard by all processors) if and only if exactly *one* processor sends a message in that round.

Moreover we assume that the processors have no possibility of *collision detection*, that is, if more than one processor (or no processor at all) send in the same round, then nothing is heard by the other processors, so making it impossible to distinguish between multi-transmission and absence of transmission. Formally, if δ is the number of processors that transmit in a given round, three cases may happen:

*Work supported in part by the European RTN Project under contract HPRN-CT-2002-00278,COMBSTRU.

†Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche, via Moruzzi 1, 56124 Pisa, Italy. E-mail: {gianluca.demarco, marco.pellegrini, giovanni.sburlati}@iit.cnr.it

‡Corresponding author.

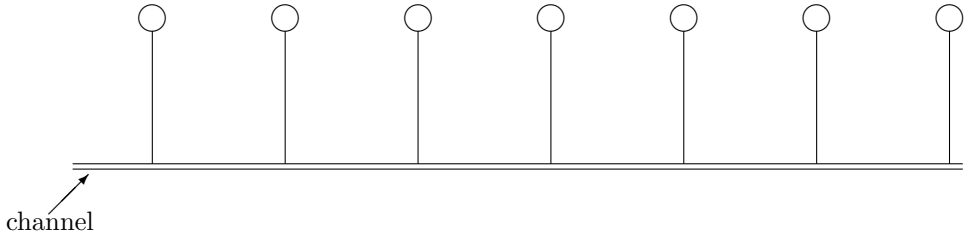

Figure 1: A multiple access channel with $n = 7$ processors.

1. $\delta = 0$: in this case, of course, no message is written on the channel;
2. $\delta = 1$: the message is written successfully on the channel and therefore heard by every processor;
3. $\delta > 1$: in this case, the δ simultaneous transmissions interfere with one another (a collision occurs) and therefore no message is written successfully on the channel.

It is clear that when no collision detection mechanism is available to the processors, cases 1. and 3. are completely undistinguishable.

A central issue is the measurement of time. Usually two extreme models are considered: the *globally synchronous* and the *locally synchronous* model. In the first model all the processors have access to a global clock. This implies that when a processor wakes up, it can see the current round number ticked by the clock. In the weaker *locally synchronous* model, each processor has its own local clock. This means that although the communication is synchronous (*i.e.* all the clocks tick with the same rate) there is no global round number. Each clock starts counting the rounds in the round in which the correspondent processor wakes up, without knowing anything about the other round numbers. In this paper, we adopt the weaker locally synchronous model.

Finally a crucial assumption is whether the processors using the shared channel are aware of the total number n of processors sharing the channel, or some polynomially related upper bound to such number. When such number n is known, much faster algorithms are possible. In keeping the line of using the weakest possible model, we assume such number to be unknown to the processors.

1.1 The problem

One of the major problems in distributed computing is the wakeup problem that consists of having a subset of processors that spontaneously awakes and has to inform all the other processors about the beginning of the computation. This problem assumes particular importance in the locally synchronous model. Indeed, as already mentioned, once one of the processors manage to send successfully its message on the channel, the message is heard by all other processors. This characteristic can be exploited to allow the processors of a locally synchronous model to switch

to the often more desirable globally synchronous one. In fact, assume that at a given round all processors hear the message sent by one of them. Starting from this round, all processors can begin to count the successive rounds with the same numbers $1, 2, ...$ That means a global clock has been created. This allows the processors to use simple communication collision-free protocols such as *time division multiplexing* TDM, in which the principle is to assign each processor a round in which the processor can transmit (which is possible only if a global clock is available).

An algorithm for the wake up problem is a collection of n transmission schedules, one for each processor, such that one of the processors eventually transmits successfully to the channel, therefore waking up every processor. This will happen in the first round at which exactly one processor sends a message.

The problem is to design an algorithm that wakes up the system as fast as possible. The difficulty comes from the fact that the algorithm has no control on the processors that *spontaneously* can wake up at any moment during the execution, therefore disturbing the communication. In other words, we assume to be playing against an adversary who controls which processors wake up and when. The time complexity is measured by the time elapsed from the round in which the first processor woke up to the round in which a processor sends successfully on the channel, therefore waking up all the system.

The wakeup problem addressed in this paper, as will be discussed in the next subsection, has been introduced in [13] and can be stated formally as follows.

Definition 1.1 (Wakeup Problem) *We are given a multiple access channel with n processors, where the parameter n is* not *known by the processors. Assume there is no global clock (locally synchronous model) and no collision detection mechanism. Suppose that processors* spontaneously and independently *can wake up at any moment. Let τ_0 be the first time slot such that some processor is woken up.*
Assign n transmission schedules, one for each processor, such that there exists a time slot τ_1 such that exactly one processor (among the awaken processors at τ_0) sends a message at τ_1. The aim is to minimize $\tau_1 - \tau_0$.

1.2 Related work

Multiple access channels includes many network systems such as Aloha multi-access systems, local area Ethernet networks, satellite communication systems, packet radio networks that have been studied extensively in the literature [2, 22]. In some of these models it is frequently assumed that a collision detection mechanism is available. As already mentioned, such a tool allows the transmitting processors to detect if its message has collided and therefore simplifies the wakeup problem. When collision detection is not available, the collisions must be *resolved* (*collision resolution*) by establishing a schedule of access to the shared communication channel that allows messages to be successively sent on the channel with as little delay as possible.

Collision detection and resolution, and access management algorithms were studied mainly assuming a known probability distribution on the arrival rate of messages at the different processors

(see *e.g.* [11, 12, 16]). Moreover, in these contexts the wakeup problem, as defined in this paper, is not considered.

As already anticipated, when collision resolution is not available, but there is a global clock, one of the simplest schedule to resolve conflicts is the time division multiplexing protocol. This means that when there are n processors, then n time slots will be needed. This becomes very inefficient when the number of awaken processors is very small compared to n.

Komlós and Greenberg [20] were the first to considered a typical situation when a subset of k among n processors are awaken and have messages, and all of them need to be sent to the channel successfully as soon as possible. The fact that in their case *all of the messages* must be sent on the channel (contrasted with our wake up problem) does not mean a real difference with our situation, since their algorithm, stopped at the first successful message sent, is actually a wake up algorithm. On the other hand, there are strong differences with our case given by the fact that in [20] the number k of awaken processors is fixed and a global clock is available. They showed how to solve the problem deterministically in time $\Omega(k + k\log(n/k))$, where either n or k is known. A lower bound of $\Omega(k(\log n)/(\log k))$ was then proved by Greenberg and Winograd [14].

The work of [20] is, to the best of our knowledge, the first that shows how to exploit *deterministically* the fact that any processor that has already transmitted successfully its message on the channel, will not transmit in the subsequent time slots, therefore avoiding to interfere with the other processors that still have to transmit.

The issues discussed in [20] are also important for their relations with Coding Theory, in particular with combinatorial structures like superimposed codes [10, 21] and its applications to combinatorial group testing. The reader interested can refer to the excellent book of Du and Hwang [8] (specially Chapter 4 and 5) for a more detailed study of the implications involved, and to the works of Indyk [17] and De Bonis and Vaccaro [6] for recent developments and generalizations.

Gąsieniec, Pelc and Peleg [13] were the first to consider the problem of waking up a multiple access channel with n processors in the synchronous setting when the number of awaken processors is not fixed, but can be any non-decreasing function of the time. In [13] many variations of synchrony and knowing assumptions are studied.

The authors of [13] considered both deterministic and randomized algorithms. For the deterministic case, which is the topic of the present paper, their result can be summarized as follows. In the globally synchronous model, when the size n is known to processors, they show an optimal deterministic algorithm that in time n solves the wakeup.
In case of unknown n, they construct a deterministic wakeup algorithm working in time $4n$ in the worst case.
Under the locally synchronous model with known n, they provide a deterministic $O(n^2 \log n)$ algorithm. They also show that even when n is known, every deterministic algorithm requires time at least $(1 + \epsilon)n$, for some $\epsilon > 0$, in the worst case.
In the locally synchronous model when n is unknown, they propose a deterministic algorithm

working in time $O(n^4 \log^5 n)$.

As for the randomized solutions (not discussed here), recent important improvements have been provided by Jurdiński and Stachowiak [19].

The wake up problem has been studied also in multi-hop radio networks, *i.e.* networks such that collisions can occur at any node in the communication graph (the multiple access channel models a single-hop radio network). Recent developments for the wakeup in multi-hop radio networks can be found in [3, 4, 5], where the authors consider the locally synchronous model when the nodes know the size n of the network (or a linear upper bound on it), but the topology is unknown. Again, knowing n is critical for the performance of such mechanisms in multi-hop networks.

1.3 Our result

We consider the wakeup problem on multiple access channel in the weakest model for the deterministic setting: locally synchronous model with unknown n. We propose a new deterministic algorithm that completes the wakeup in time $O(n^4 \log^3 n)$ in the worst case. This is an improvement over the previous $O(n^4 \log^5 n)$ algorithm presented in [13] and it is based on a different approach. Since the bound $O(n^4 \log^3 n)$ depends on a result in Number Theory (Theorem 2.3 below) which probably is not tight (see our remark at the end of Section 4), actually the bound provided by our new algorithm might be improved.

2 Preliminaries

In this section we recall some well known results in Number Theory that will be used in the analysis of our algorithm. Throughout the paper the ith prime number will be denoted p_i. The prime counting function $\pi(n)$ is the function giving the number of primes $\leq n$.

The Prime Number Theorem gives an asymptotic form for the prime counting function.

Theorem 2.1 (Prime Number Theorem)

$$\pi(n) \sim \frac{n}{\ln n}.$$

(Equivalently, $p_n \sim n \ln n$*).*

□

We have used the asymptotic notation $\sim$ as defined in [15]: in other words, the Prime Number Theorem tells us that the limit of the quotient of the two functions $\pi(n)$ and $n/\ln(n)$, as n approaches to infinity, is 1.

The theta function $\theta(n)$ is defined as follows:

$$\theta(n) = \sum_{i=1}^{\pi(n)} \ln p_i = \ln \left(\prod_{i=1}^{\pi(n)} p_i \right).$$

Chebyshev [15, p. 341] gave the following bound for $\theta(n)$.

Theorem 2.2 (Chebyshev's bound on $\theta(n)$)

$$\theta(n) < 2n \ln 2 \text{ for all } n \geq 1.$$

□

Jacobsthal's problem is to estimate, for a given r, the maximum length of a sequence of consecutive integers, each divisible by one of r arbitrarily chosen primes. For references and history on this problem, see [9]. Iwaniec [18] proved the following important result.

Theorem 2.3 (Iwaniec, 1978) *For any positive integer r, let $C(r)$ denote the maximal length of a sequence of consecutive integers each divisible by at least one of r arbitrarily fixed primes. Then*

$$C(r) \in O(r^2 \ln^2 r) \quad (r \to +\infty).$$

□

We will also use the following well known result.

Theorem 2.4 (Chinese Remainder Theorem) *Let $m_1, \ldots, m_r$ be pairwise relatively prime positive integers. Let $M = m_1 \cdots m_r$ and let $a_1, \ldots, a_r, A$ be integers. Then there is exactly one integer a such that $A \leq a < A + M$ satisfying*

$$a \equiv a_k \pmod{m_k} \quad \forall\, k,\ 1 \leq k \leq r.$$

□

3 Wakeup in the locally synchronous model with unknown n

Here we consider the weakest model: there is no global clock available and the size n of the system is not known; each processor knows its own ID number, all ID numbers are distinct.

3.1 The new upper bound

Algorithm FAST_WAKEUP**:** *For each $j \in \mathbb{N}$, let p_j be the j-th prime number. Each processor i transmits immediately when it is woken up, and successively it transmits exactly p_i time units after its last transmission.*

Theorem 3.1 *Algorithm* FAST_WAKEUP *wakes up n processors in time $O(n^4 \ln^3 n)$.*

Proof: In order to measure the time necessary to wake up the system, we will often refer our calculation to a global time t to mean the tth time unit after the first processor has woken up. Of course, this time (unknown to the processors) does not have to be confused with the local times. For each integer i with $1 \leq i \leq n$, a_i is the time at which the processor labelled i wakes up ($a_i = +\infty$ if the i-th processor does not wake up during the whole process).

We call W the set of the labels of the processors that wake up in the process (*i.e.* $W = \{i : (1 \leq i \leq n) \wedge (a_i < +\infty)\}$).

Following the algorithm, it is easy to verify that any processor i transmits at a generic time $x \geq a_i$ if and only if $x \equiv a_i \pmod{p_i}$.

Let a_k (with $1 \leq k \leq n$) be a generic time unit at which some processor k wakes up; let m be the maximal label among the processors that are awake at time a_k, that is, $m = \max\{i : (1 \leq i \leq n) \wedge (a_i \leq a_k)\}$. We want to find a time unit $x \geq a_k$ at which processor k transmits and for any $i \in W$, $i \neq k$, $1 \leq i \leq m$ processor i does not transmit. A sufficient condition for this to happen is

$$\begin{cases} x \equiv a_k & (\text{mod } p_k), \\ x \not\equiv a_i & (\text{mod } p_i) \quad \forall\, i \in W,\, i \neq k,\, 1 \leq i \leq m. \end{cases}$$

Let us set $N = p_1 p_2 ... p_m$. From Theorem 2.2 it follows that $\sum_{1 \leq i \leq m} \ln p_i < p_m \ln 4$. Therefore, $N < 4^{p_m}$ and, since by Theorem 2.1 $p_m \in O(m \ln m)$, we must have $N \in 4^{O(m \ln m)}$. By Theorem 2.4 there exists at least a y in the range $1 \leq y \leq N$ satisfying

$$y \equiv a_k - a_i \pmod{p_i} \quad \forall\, i \in W,\, 1 \leq i \leq m; \tag{1}$$

in particular for $i = k$ we obtain $y \equiv a_k - a_k \equiv 0 \pmod{p_k}$.

Consider the integer $y' = y/p_k$; by theorem 2.3 there exists a non-negative $\ell' \in O(m^2 \ln^2 m)$ such that $y' + \ell'$ is an integer, say r, which is co-prime to N. We can then write

$$y/p_k + \ell' = r, \text{ with } \ell' \in O(m^2 \ln^2 m) \text{ and } \gcd(r; N) = 1. \tag{2}$$

Let $\ell = p_k \ell'$, multiplying the two members of (2) by p_k gives $y + \ell = p_k r$. Since $p_k \leq p_m$, p_k lies in $O(m \ln m)$. Therefore,

$$\ell \in O(m \ln m) \cdot O(m^2 \ln^2 m) = O(m^3 \ln^3 m).$$

Since y satisfies all the congruences of system (1), from the equality $y + \ell = p_k r$ we can deduce that

$$\forall\, i \in W \text{ with } 1 \leq i \leq m,\ a_k - a_i + \ell \equiv p_k r \pmod{p_i},$$

i.e.

$$\forall\, i \in W \text{ with } 1 \leq i \leq m,\ a_k + \ell \equiv a_i + p_k r \pmod{p_i}. \tag{3}$$

Recalling that $\gcd(r; N) = 1$ (and then p_i does not divide r for any i with $1 \leq i \leq m$), the congruences in (3) imply

$$\begin{cases} a_k + \ell \equiv a_k & \pmod{p_k}, \\ a_k + \ell \not\equiv a_i & \pmod{p_i} \quad \forall\, i \in W,\ i \neq k,\ 1 \leq i \leq m. \end{cases}$$

This means that, once processor k wakes up at time a_k, it transmits successfully within ℓ time units (where $0 \leq \ell \in O(m^3 \ln^3 m) \subseteq O(n^3 \ln^3 n)$), unless in the meantime another processor has been woken up after it. Since at most n processors can be woken up, it follows that a successful transmission will be definitely produced within $n \cdot O(n^3 \ln^3 n) = O(n^4 \ln^3 n)$ time units after any processor wakes up.

□

4 Conclusion and further research

In this paper, we have showed an upper bound of $O(n^4 \ln^3 n)$ on the time for waking up deterministically a locally synchronous multiple access channel of n processors, when n is unknown to the processors. The best lower bound for this problem is $(1 + \epsilon)n$, for some $\epsilon > 0$, proved in [13].

Hence, the main question left open by the present paper is to narrow the (still huge) gap between upper and lower bound. To this aim it is useful to remark that our $O(n^4 \ln^3 n)$ upper bound given in Theorem 3.1 depends essentially on the $O(r^2 \ln^2 r)$ upper bound of Theorem 2.3. Such an upper bound probably is not optimal: in [18] Iwaniec cites the open question raised by Jacobsthal about the validity of the stronger upper bound $O(r^2)$. If this latter bound holds, the arguments used in the proof of Theorem 3.1 would lead to the result $O(n^4 \ln n)$ instead of $O(n^4 \ln^3 n)$.

Acknowledgements

We wish to thank Jean-Louis Nicolas for his guidance on the literature about Jacobstahl's problem.

References

[1] R. Bar-Yehuda, O. Goldreich and A. Itai, *On the Time-Complexity of Broadcast in Multi-hop Radio Networks: An Exponential Gap between Determinism and Randomization.* Journal of Computer and System Sciences Vol. 45, 1992, pp. 104-126.

[2] D. Bertsekas and R. Gallager, *Data Networks.* Prentice Hall, 1991.

[3] M.Chrobak, L.Gasieniec and D.Kowalski *The Wake-Up problem in multi-hop radio networks*, In Proceedings of 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2004), pp. 992-1000.

[4] B. Chlebus, L. Gasieniec, D. Kowalski and T. Radzik *On the Wake-up Problem in Radio Networks*, to appear in the proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP 2005).

[5] B. Chlebus and D. Kowalski *A better wake-up in radio networks*, Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing (PODC 2004), pp. 266-274.

[6] A. De Bonis and U. Vaccaro, *Efficient Constructions of Generalized Superimposed Codes with Applications to Group Testing and Conflict Resolution in Multiple Access Channels*, Theoretical Computer Science, vol. 306, Issue 1-3, pp. 223-243, 2003.

[7] G. De Marco and A. Pelc, *Faster broadcasting in unknown radio networks,* Information Processing Letters Vol 79, 2001, pp. 53-56.

[8] D.Z. Du and F.K. Hwang, *Combinatorial Group Testing and its Applications*, World Scientific, 2000.

[9] P. Erdös, *On the Integers Relatively Prime to n and on a Number-Theoretic Function Considered by Jacobsthal,* Math. Scand. 11 (1962), pp. 163-170.

[10] P. Erdös, P. Frankl and Z. Füredi, *Families of Finite Sets in Which no Set Is Covered by the Union of r Others,* Israel J. Math., vol. 51, 1985, pp. 75-89.

[11] R. Gallager, *A Perspective on Multiaccess Channels,* IEEE Trans. on Information Theory 31 (1985), pp. 124-142.

[12] J. Goodman, A.G. Greenberg, N. Madras, P. March, *On the stability of Ethernet*, 17th ACM symposium on Theory of computing, STOC, 1985, pp. 379-387.

[13] L. Gąsieniec, A. Pelc and D. Peleg, *The Wakeup Problem in Synchronous Broadcast Systems.* SIAM J. Discrete Math., Vol 14, No. 2, 2001, pp. 207-222.

[14] A.G. Greenberg and S. Winograd, *A Lower Bound on the Time Needed in the Worst Case to Resolve Conflicts Deterministically in Multiple Access Channels.* J. ACM, 32 (1985), pp. 598-596.

[15] G.H. Hardy and E.M. Wright, *An Introduction to the Theory of Numbers*, 4th ed. Oxford, England: Clarendon Press, 1975.

[16] J. Hastad, T. Leighton, B. Rogoff, *Analisys of Backoff Protocols for Multiple Access Channels,* 19th ACM symposium on Theory of computing, STOC, 1987, pp. 241-253.

[17] P. Indyk, *Deterministic Superimposed Coding with Application to Pattern Matching,* 38th Symposium on Foundations of Computer Science, FOCS, 1997, pp. 127-136.

[18] H. Iwaniec, *On the problem of Jacobstahl*, Demonstratio Mathematica, **11** (1978), 225-231.

[19] T. Jurdziński and Stachowiak *Probabilistic Algorithms for the Wakeup Problem in Single-Hop Radio Networks*, ISAAC 2002, LNCS 2518, pp.535-549, 2002.

[20] J. Komlós and A.G. Greenberg, *An Asymptotically Optimal Nonadaptive Algorithm for Conflict Resolution in Multiple-Access Channels*, IEEE Trans. on Information Theory, vol. 31, (2), (1985), pp. 302 - 306

[21] W.H. Kautz and R.C. Singleton, *Nonrandom binary superimposed codes,* IEEE Trans. Inform. Theory, vol. 10, 1964, pp. 363 - 377.

[22] A.Tannenbaum, *Computer Networks,* Prentice-Hall, Englewood Cliffs, NJ, 1981.