On the Sparse Set Conjecture for Sets with Low Density

Harry Buhrman
Montserrat Hermo

Report LSI-94-23-R
On the Sparse Set Conjecture for Sets with Low Density
(Extended Abstract)

Harry Buhrman* Montserrat Hermo†

June 21, 1994

Abstract

We study the sparse set conjecture for sets with low density. The sparse set
conjecture states that $P = NP$ if and only if there exists a sparse Turing hard set for
$NP$. In this paper we study a weaker variant of the conjecture. We are interested in the
consequences of $NP$ having Turing hard sets of density $f(n)$, for (unbounded) functions
$f(n)$, that are sub-polynomial, for example $\log(n)$. We establish a connection between
Turing hard sets for $NP$ with density $f(n)$ and bounded nondeterminism: We prove
that if $NP$ has a Turing hard set of density $f(n)$, then satisfiability is computable in
polynomial time with $O(\log(n) \cdot f(n^c))$ many nondeterministic bits for some constant
c. As a consequence of the proof technique we obtain absolute results about the density
of Turing hard sets for $EXP$. We show that no Turing hard set for $EXP$ can have
sub-polynomial density. On the other hand we show that these results are optimal
w.r.t. relativizing computations. For unbounded functions $f(n)$, there exists an oracle
relative to which $NP$ has a $f(n)$ dense Turing hard tally set but still $P \neq NP$.

1 Introduction

The density of $NP$ -complete and hard sets has been an early object of study and starts
with the seminal paper of Berman and Hartmanis [BH77]. In that paper roughly two lines
of research have been initiated: the density of $\leq_p^n$ -complete sets and the density of $\leq_p^T$
-hard sets for $NP$ and other complexity classes.

The study of the $\leq_p^n$ -complete sets for $NP$ becomes apparent in relation with the
isomorphism conjecture [BH77]. The conjecture implies that all $NP$ -complete sets are
exponentially dense, since SAT, the well known $NP$ -complete set, itself is of exponential
density. Along these lines it was Mahaney who showed that $NP$ -complete sets can not
be polynomially dense unless $P = NP$.

The study of the $\leq_p^n$ -hard sets for $NP$ is motivated by the equivalence between sets
that are $\leq_p^n$ -reducible to a sparse set, sets that have polynomial size circuits [BH77] and
sets that can be recognized in polynomial time with the additional help of a polynomial

*Dept. Llenguatges i Sist. Informàtiques Univ. Politècnica de Catalunya Pau Gargallo 5, 08028 Barcelona, Spain. E-mail: harry@goliat.upc.es. Supported by a TALENT stipendium from the Netherlands Organization for Scientific Research (NWO)
†Dept. de Lenguajes y Sist. Informáticos Universidad del País Vasco E-20080, San Sebastian, Spain. E-mail: jiphehum@si.ehu.es. Part of this work was done while visiting UPC Barcelona
amount of advice \((P/poly)\) [Pip79]. Hence if there exists a \(\leq_T^P\) -hard sets for \(NP\) that is sparse, there exists a polynomial time algorithm for SAT that needs the help of a small (polynomial size) table. For practical purposes this would mean that one only had to compute this small table once (for inputs of a certain length) and that from then on \(NP\) would equal \(P\) for all inputs of this length. Karp and Lipton however showed that the existence of sparse \(\leq_T^P\) -hard sets for \(NP\) implies an unlikely collapse of the Polynomial Time Hierarchy to its second level [KL80].

Many efforts have been put into improving the Karp and Lipton result to \(P = NP\). This improvement also goes by the name "sparse set conjecture". An important step towards this conjecture was obtained by Ogihara and Watanabe, who showed that indeed the stronger consequence, i.e. \(P = NP\), can be obtained if \(\leq^P_{bit}\) -reductions are used instead of \(\leq_T^P\) -reductions [OW90].

Not only does the sparse set conjecture imply a better understanding of the structure of \(NP\) and \(NP\) -hard sets, also it does settle the blatant inability to prove that \(EXP\) does not have polynomial size circuits. Best known upper bounds on this can be found in [Kan82].

Attempts to prove the sparse set conjecture usually result in studying stronger than Turing reducibility types to sparse sets [OW90, HL91, OL91, AHH+91]. In this paper we follow a different line of attack. Instead of strengthening the reduction type we study the most general Turing reduction and vary the density of the set reduced to. In particular are we interested in the consequences of the existence of \(\leq_T^P\) -hard sets for \(NP\) that have smaller density than sparse sets. The analogous question for sets with bigger than polynomial, i.e. super-polynomial, density has been addressed in [BH92].

We study the consequences of the existence of \(\leq_T^P\) -hard sets for \(NP\), that contain only \(f(n)\) strings of size less than or equal to \(n\), for \(f(n)\) a unbounded function that is strictly smaller than any polynomial. We show that there is a link between classes of bounded nondeterminism [KF80, DT90], and \(NP\) having hard sets with low density. We prove that under the assumption that \(NP\) has a \(\leq_T^P\) -hard set with density \(f(n)\), SAT can be computed with only \(O(\log(n) \cdot f(n^c))\) bits of nondeterminism, for some constant \(c\). Taking for \(f(n)\) for example \(\log(n)\) results in a collapse of \(NP\) to the second level of the Beta Hierarchy[DT90]. This on its turn implies for example that \(EXP = NEXP\).

Note \(P\) can be characterized as the class of sets that are recognized by nondeterministic polynomial time machines that use \(O(\log(n))\) bits of nondeterminism.

On the other hand we show that this result is optimal with respect to relativizing computations, even if we consider tally sets. We prove that there exists an oracle relative to which \(NP\) does have Turing hard tally sets with \(f(n)\) density but \(P \neq NP\), for \(f(n)\) an unbounded function.

As an application of the developed proof technique we establish absolute results concerning the density of \(\leq_T^P\) -hard sets for \(EXP\). We prove that Turing hard sets for \(EXP\) can not be of sub-polynomial density. This is in some sense optimal, since improvement to polynomial density, would show that \(EXP\) does not have polynomial size circuits, and Wilson [Wil85] showed the existence of an oracle where \(EXP\) does have polynomial size circuits.

The main results in this paper are:

- We establish a connection between classes of bounded nondeterminism and \(NP\) hav-
ing Turing hard sets with low density.

- We prove that if \( NP \) has a Turing hard set with density \( f(n) \), for any unbounded function \( f(n) \), then SAT can be computed in polynomial time with the use of \( O(\log(n) \ast f(n^c)) \) many nondeterministic bits, for \( c \) some constant.

- We show that these results are optimal with respect to relativized computations: there exists an oracle relative to which \( NP \) has a Turing hard tally set with density \( f(n) \), but \( P \neq NP \).

- As a consequence of the developed proof technique we show that Turing hard sets for \( EXP \) can not have sub-polynomial density.

The results suggest that it is probably hard to prove the sparse set conjecture even if we consider sets with arbitrary low density. On the other hand this line of research might give a handle on proving the actual sparse set conjecture. It seems more doable to work on non-relativizing proof techniques for proving the sparse set conjecture, for sets with low density than for sparse sets.

2 Preliminaries

We assume the reader familiar with standard notions in structural complexity theory, as are defined e.g. in [BDG88]. We will be using (non)deterministic polynomial time oracle Turing machines. Let \( M \) be a (non)deterministic polynomial time oracle Turing machine. We will denote \( Q(M, x, A) \) as the set of queries \( M \) makes on input \( x \) with oracle \( A \). Note that if \( M \) is a deterministic machine then \( |Q(M, x, A)| \) is bounded by a polynomial in the length of \( x \).

Apart from SAT, the well known \( NP \) -complete set, we will be using the set \( K^A \) as well. Where \( K^A \) is defined as follows:

\[
K^A \subseteq \{<i, x, 0|x|i> | M^A(x) \text{ accepts } x \text{ within } |x|i \text{ steps} \}
\]

We will be considering efficient reductions of complete sets for various classes to sets of different subpolynomial densities. The following definition specifies our measure of a set’s density.

**Definition 1** Let \( f \) be a nondecreasing function, \( f : \mathbb{N} \to \mathbb{N} \). A set \( S \subseteq \Sigma^* \) is \( f(n) \) dense or has \( f(n) \) density if \( |S^{\leq n}| < f(n) \) for all \( n \). If \( F \) is a class of functions, we say \( S \) is \( F \) dense if \( S \) is \( f \) dense for some \( f \in F \).

\( S \) is said to be sparse if \( S \) is \( P \) dense. Where \( P \) is the class of all polynomials. \( S \) is said to be of sub-polynomial density if \( S \) is \( F \) dense, where \( F \) is a class of functions such that \( f \in F \) iff \( \forall c \exists n_0 \forall n : f(n) < n_0 \). We will call such a function sub-polynomial.

We will be using classes that are defined by limiting the number of nondeterministic moves of a nondeterministic polynomial time Turing machine [KF80, DT90].

**Definition 2** \([DT90]\) For any function \( f : \mathbb{N} \to \mathbb{N} \) let

\[
\beta_f = \{L \mid \exists A \in P, \exists c, \forall x : x \in L \iff (\exists y, |y| \leq c \ast f(|x|) \text{ and } <x, y> \in A)\}
\]
Will will also use the notation $\beta^a_f$ for the class of sets recognizable by a polynomial time oracle Turing machine that uses at most $f(n)$ nondeterministic moves and has a set in $\beta_f$ as oracle.

3 Hard sets for NP and Bounded Nondeterminism

In this section we study the consequences of NP having Turing hard sets with low density. We are interested in sets with sub-polynomial density.

Let us consider an example of such a set. Let $f(n) = \log(n)$. Consider the assumption that $NP$ has a Turing hard tally set $T$ of density $\log(n)$. This means that SAT $\leq^p_T T$, say in time $n^c$. The following counting argument, together with a nowadays standard technique, yield that SAT is computable in time $DTIME(c' \cdot n^{\log(n)})$, for some constant $c' = c^2$.

First lets count how many different tally sets $T \leq n^c$ of density $\log(n)$ there exist. Each $T \leq n^c$ may contain at most $c \cdot \log(n)$ strings, that can be placed at $n^c$ many different positions, hence the number of different tally sets up to length $n^c$ is bounded by:

$$\binom{n^c}{c \log(n)} \leq n^{c' \cdot \log(n)} \text{, where } c' = c^2$$

Next we will use the fact that it is possible to compute in polynomial time relative to SAT, for any satisfying formula $\phi$, using the disjunctive selfreducibility of SAT, a satisfying assignment for $\phi$. This property is also called Search Reduces to Decision or Functional Selfreducibility [BD76, BBFC91, NOS93]. Since SAT reduces to some tally set with $\log(n)$ density, it is reducible to one among the $n^{c'=\log(n)}$ many different ones. Consider the following algorithm. On input $\phi$ cycle through all the possible tally sets of $\log(n)$ density. For each one try to compute a satisfying assignment for $\phi$ and accept if and only if one is found. It is clear that a satisfying assignment will be found this way if and only if $\phi \in$ SAT. Furthermore this procedure runs in time $n^{c' \cdot \log(n)}$.

The problem with the previous approach is that it does not work for sets over $\{0, 1\}$. The number of possible sets with density $\log(n)$ is only bounded by $2^{n^{c'=\log(n)}}$ and hence will yield that SAT is computable in time $2^{n^{c'=\log(n)}}$, but this is not very good since SAT is computable in time $2^n$. The following theorem shows that linking the problem to bounded nondeterminism yields even better results than the above approach for non tally sets. We will show that for any function $f(n)$ the assumption that $NP$ has a Turing hard set with density $f(n)$ implies that SAT is computable in polynomial time with $O(\log(n) \cdot f(n^c))$ many nondeterministic bits, for some constant $c$.

**Theorem 3** Let $f(n)$ be any fully time constructible function. If there exists a set $S$ with density $f(n)$ that is Turing hard for NP then SAT is computable in polynomial time with $O(\log(n) \cdot f(n^c))$ many nondeterministic bits, for some constant $c$.

**Proof:** (Sketch) We have to show that under the assumption that SAT reduces to a set $S$ with density $f(n)$ we can construct a polynomial time algorithm using $O(\log(n) \cdot f(n^c))$ many nondeterministic bits, that decides SAT. Assume that SAT $\leq^p_T S$ via a machine $M_c$, that runs in time $n^c$ and that $M_a$ witnesses the fact that SAT has Search Reducing to Decision. Simulate the machine $M_a$, that generates a satisfying assignment relative to
SAT on input $\phi$. Every time $M_a$ makes a query to SAT simulate the Turing reduction $M_c$, from $A$ to $S$, on this query. Every time $M_c$ makes a query $q$ to $S$ either assume that $q$ is out of $S$ or guess that $q$ is in $S$. Make sure that at most $f(n^c)$ times a query is guessed to be in $S$ and that the decision about $q$ is consistent with previously made decisions about $q$. Accept if and only if a satisfying assignment for $\phi$ is found. It should be clear that this construction runs in polynomial time. From this it follows that at most $O(\log(n) * f(n^c))$ many nondeterministic bits are used. The following describes in more detail the above algorithm:

\textbf{input} $\phi$
\textbf{\texttt{n}} := $|\phi|$
Guess a set \textit{Positions} of size $\leq f(n^c)$ such that $\forall i \in \text{Positions}$, $|i| \leq \log(n^{c+1})$
$\text{Queries} := \emptyset$
$\text{Count} := 0$
$z := \phi$

\textbf{while} there is at least one variable in $z$
\text{let} $x$ be the first variable in $z$
$z' := z|_{x:=0}$

\textbf{while} Simulation of $M_c$ on $z'$ not ended \textbf{do}
\text{Simulate} $M_c$ on $z'$ until it queries $q$
$\text{Count} := \text{Count} + 1$
\textbf{if} $q \in \text{Queries}$ \textbf{then} continue simulation in YES state
\textbf{else if} $\text{Count} \in \text{Positions}$ \textbf{then}
\text{Queries} := \text{Queries} \cup \{q\}$
continue simulation in YES state
\textbf{else} continue simulation in NO state
\textbf{end while}
\textbf{if} $M_c$ accepted $z'$ \textbf{then} $z := z'$
\textbf{else} $z := z|_{x:=1}$
\textbf{end while}
\textbf{if} $z$ simplifies to "true" \textbf{then} accept \textbf{else} reject

\hfill \Box

The above Theorem shows that although a set $S$ over \{0, 1\} with density $f(n)$ (for $f(n)$ small) contains in some sense more information than a tally set $T$ of the same density, a polynomial time algorithm is not able to extract this information out of $S$.

Plugging in explicit values for $f(n)$ yields the following corollary:

\textbf{Corollary 4} If there exists a set $S$ with density $\log(n)^k$, that is Turing hard for NP, then:

1. $NP = \beta_{\text{polylog}}$
2. $EXP = NEXP$.

\textbf{Proof:} Use Theorem 3 together with standard padding arguments. \hfill \Box

5
Another consequence of the proof technique of Theorem 3 is that it yields absolute results about the density of Turing hard sets for \( EXP \). To our knowledge this is the first result concerning the density of \( \leq^p_T \) -hard sets for \( EXP \).

**Theorem 5** There do not exist Turing hard sets for \( EXP \) that have sub-polynomial density.

**Proof:** (Sketch) Assume that there exists a \( f(n) \) dense Turing hard set for \( EXP \) for some sub-polynomial function \( f \). From this we can conclude [KL80] that \( EXP = \Sigma^p_2 \). We will see that by extending Theorem 3 we will be able to show that \( \Sigma^p_2 \) is computable in time \( 2^{O(\log(n)^{f(n^O(1))})} \). Then we will have a contradiction with the hierarchy Theorems for deterministic time, since \( EXP \) is not computable in sub-exponential deterministic time. In order to extend Theorem 3 we will make use of the fact that all \( \leq^p_T \) -complete sets for \( \Sigma^p_2 \) are functional selfreducible [BD76]. Functional selfreducibility is the natural generalization of search reducing to decision for other levels of the Polynomial Time Hierarchy. Consider any set \( A \) in \( \Sigma^p_2 \). By definition there exists a polynomial time computable relation \( R_A \) such that:

\[
x \in A \iff \exists y \forall z : R_A(x, y, z)
\]

A set \( A \Sigma^p_2 \) is functional selfreducible if there exists a polynomial time procedure that can compute, relative to \( A \), for all \( x \in A \) a \( y_0 \) such that \( \forall z : R_A(x, y_0, z) \). For all strings not in \( A \) such a \( y_0 \) does not exists. Borodin and Demers showed that this is always the case if \( A \) is \( \leq^p_T \) -complete for \( \Sigma^p_2 \) [BD76]. We will see that under the assumption of the existence of a \( f(n) \) dense hard set for \( EXP \) any complete set \( A \) for \( \Sigma^p_2 \) can be recognized in \( \beta^p_{g(n)} \), for \( g(n) \in O(\log(n)^{f(n^O(1))}) \).

Let \( A \) be \( \leq^p_T \) -complete for \( \Sigma^p_2 \), \( M_f \) be the procedure that witnesses that \( A \) is functional selfreducible and \( M_c \) be the machine witnessing that \( A \leq^p_T S \). Simulate \( M_f \) on some input \( x \) and every time a query to \( A \) is made simulate the reduction \( M_c \), from \( A \) to \( S \), on this query. Again, as in the proof of Theorem 3, if \( M_c \) makes a query \( q \) to \( S \) assume either that \( q \) is not in \( S \) or guess that the query is in the set, making sure that no more than \( f(n^c) \) many yes guesses are made and that the decisions are consistent with previously made decisions about \( q \). After this computation some \( y' \) has been computed. If nothing is computed, because \( y_0 \) does not exists or because some wrong sequence of guesses is used, reject. Next the algorithm checks whether the computed \( y' \) indeed satisfies the property that \( \forall z : R_A(x, y', z) \). This can be done by one oracle call to SAT. Since \( SAT \in \Sigma^p_2 \) it follows using Theorem 3 that \( SAT \in \beta^p_{O(\log(n)^{f(n^d)})} \) for some \( d \). Hence \( A \in \beta^p_{g(n)} \) and can be decided in deterministic sub-exponential time.

Note that any improvement to the previous theorem will settle the big the open question whether \( EXP \) has polynomial size circuits. Also does the theorem hint at the possibility that a much weaker form of the sparse set conjecture, might settle the circuit issue for \( EXP \).

---

1We mean here the following weaker variant: SAT is computable in polynomial time with significantly less than a linear number of nondeterministic bits if and only if there exists a sparse Turing hard set for \( NP \).
4 Relativized Optimality

In this section we show that Theorem 3 is optimal with respect to relativized computations.

**Theorem 6** Let $f(n)$ be an unbounded function. There exists an oracle $A$ and a tally set $T$ of density $f(n)$ such that:

1. $K^A \leq_T P^A T$ and
2. $P^A \neq NP^A$.

**Proof:**

(Sketch) W.l.o.g. we assume that $f(n) < n$ and monotone. The construction of the set $A$ consists out of two parts. One to satisfy requirement 1 and the other part to satisfy 2. The set $A$ will be the marked union of QBF, a $PSPACE \leq_T P_m$-complete set, and a set $B$ (i.e. $A = \text{QBF} \oplus B$). Requirement 2 will be satisfied by showing that the following test language $L$ is in $NP^A$ but not in $P^A$:

$$L = \{0^n \mid \exists x \in B \text{ and } |x| = n\}$$

Note that for all $B$, $L$ is in $NP^A$. We will construct $B$ in such a way that $L \notin P^A$. In order to do this we need a sequence of strings to diagonalize over. Let $\{a_i\}_{i=1}^{\infty}$ be a sequence of natural numbers such that the following things hold:

1. $f(a_i) \geq 2 * f(a_{i-1})$,
2. $f(a_{i-1}) * a_{i-1} < a_i$,
3. $a_i^{f(a_i)} > a_i^i + i$, and
4. $a_0$ is the first $n$ such that $f(n) \geq 2$.

Since $f(n)$ is unbounded it is not hard to see that $\{a_i\}_{i=0}^{\infty}$ exists and is infinite. In the following let $A = \text{QBF} \oplus B$ and let $M_1, M_2, \ldots$ be an enumeration of polynomial time oracle Turing machines such that $M_i$ runs in time $n^i + i$. The construction of $B$ goes as follows:

**Stage $n$:**

Let $y_{a_i} = x_1 x_2 \ldots x_k 00 \ldots 0$ be a string such that:

1. $k = \log(a_i) + f(a_i)$,
2. $x_i \in \{0, 1\}$ $(1 \leq i \leq k)$,
3. $|y_{a_i}| = a_i$, and
4. $y_{a_i} \not\in Q(M_i, 0^{a_i}, A)$.

Put $y_{a_i}$ into $B$ if and only if $M_i^{A}(0^{a_i})$ rejects.

**End of stage $n$**
From the definition of \(\{a_i\}_{i=0}^{\infty}\), the fact that \(M_1\) runs in time \(n^i + i\), and that \(f(n)\) is unbounded it follows that \(y_{a_i}\) exists and that \(L \not\in P^A\). The construction of \(B\) satisfies requirement 2. In order to satisfy requirement 1 we need to code \(B\) into a \(f(n)\) dense tally set. We will show that we can code \(B\) into a \(2 \times f(n)\) dense tally set \(T\), which is sufficient\(^2\) to satisfy requirement 1. First observe the following. If we put, for all \(n\), \(f(a_n)\) strings into \(T\) at the interval \(0^{a_n}, \ldots, 0^{a_n \times f(a_n)}\) then \(T\) will have density \(2 \times f(n)\). The observation yields that we have \(f(a_n)\) strings to code \(y_{a_n}\) into \(T\). To do this divide the interval \(0^{a_n}, \ldots, 0^{a_n \times f(a_n)}\) into \(f(a_n)\) intervals \(I_k\) of length \(a_n\) such that \(I_k = 0^{k \times a_n}, \ldots, 0^{((k+1) \times a_n - 1)}\) \((1 \leq k < f(a_n))\). Next if we place one string \(t_k\) in each \(I_k\) we can interpret \(t_1t_2 \ldots t_{f(a_n)}\) as a \(f(a_n)\) digit number base \(a_n\). Recall that \(y_{a_n} = x_1x_2 \ldots x_k00 \ldots 0\) where \(k = \log(a_n) \times f(a_n)\). The only part of \(y_{a_n}\) that matters is \(x_1x_2 \ldots x_k\). Since we can express \(x_1x_2 \ldots x_k\) as a \(f(a_n)\) digit number base \(a_n\) there is room to code \(x_1x_2 \ldots x_k\) and hence \(y_{a_n}\) into \(T\).

This coding argument shows that we can code \(B\) into \(T\). To see that \(K^A \leq_P T\) we do the following reduction. Let \(M_j\) be some nondeterministic polynomial time oracle machine that accepts \(K^A\). In order to know whether \(M_j^A(x)\) accepts or rejects, the reduction firsts recovers \(B\) up to length \(|x|^j\), using \(T\). Next it queries QBF whether \(M_j(x)\) with QBF and \(B\) up to length \(|x|^j\) accepts or rejects. □

5 Conclusions

The oracle result (Theorem 6) shows that the weaker form of the sparse set conjecture needs nonrelativizing proof techniques to be proven. On the other hand it seems more doable to develop non-relativizing techniques for the weaker form of the conjecture than for the actual conjecture.

Another interesting point that comes out of the oracle construction is that the idea behind the coding can be used to show that the class of sets that Turing reduce to a tally set with polylog\(^3\) density and the nonuniform advice class \(\text{Full-P/polylog}\) \([Ko87, \text{BHM92}]\) are equal. It seems therefore natural to study the structure of these advice classes in greater detail.

Theorem 5 suggests that the weaker form of the sparse set conjecture, i.e \(NP = \beta_f(n)\) iff exists a Turing hard sparse set for \(NP\) for some subpolynomial function \(f(n)\), implies that \(EXP\) is not contained in \(P/poly\). It would be interesting to prove this.

Acknowledgements

We would like to thank José Balcazar and Ricard Gavaldà for listening to us with great patience and suggesting interesting ideas. Special thanks are due to Jacobo Torán for pointing out the importance of limited nondeterminism.

\(^2\)This actually only shows that for each unbounded function \(f(n)\) there exists a \(2 \times f(n)\) dense tally set with the desired properties. However starting out with a function \(f'(n) = f(n)/2\) will satisfy requirement 1.

\(^3\)Polylog stands for the class of functions \(\{f | \exists i, f(n) = \log^i(n)\}\).
References


LSI-94-2-R “Degrees and reducibilities of easy tally sets”, Montserrat Hermo.

LSI-94-3-R “Isothetic polyhedra and monotone boolean formulae”, Robert Juan-Ariño.

LSI-94-4-R “Una modelización de la incompletitud en los programas” (written in Spanish), Javier Pérez Campo.


LSI-94-6-R “Construction of the Face Octree model”, Núria Pla-Garcia.

LSI-94-7-R “On the expected depth of boolean circuits”, Josep Díaz, María J. Serna, Faúl Spirakis, and Jacobo Torán.

LSI-94-8-R “A transformation scheme for double recursion”, José L. Balcázar.


LSI-94-10-R “Relative knowledge and belief: SKL preferred model frames”, Matías Alvarado.

LSI-94-11-R “A top-down design of a parallel dictionary using skip lists”, Joaquim Cabarró, Conrado Martínez, and Xavier Messeguer.

LSI-94-12-R “Analysis of an optimized search algorithm for skip lists”, Peter Kirschenhofer, Conrado Martínez, and Helmut Prodinger.

LSI-94-13-R “Bases de dades bitemporals” (written in Catalan), Carme Martín and Jaume Sistac.

LSI-94-14-R “A volume visualization algorithm using a coherent extended weight matrix”, Daniela Tost, Anna Puig, and Isabel Navazo.


LSI-94-17-R "SAREL: An assistance system for writing software specifications in natural language", Núria Castell and Angels Hernández.

LSI-94-18-R "Medición del factor modificabilidad en el proyecto LESD" (written in Spanish), Núria Castell and Olga Slávkova

LSI-94-19-R "Algoritmes paral·lels SIMD d'extracció de models de fronteres a partir d'arbres octals no exactes" (written in Catalan), Jaume Solé and Robert Juan-Arinyo.

LSI-94-20-R "Una paral·lelització SIMD de la conversió d'objectes codificats segons el model de fronteres al model d'octrees clàssics" (written in Catalan), Jaume Solé and Robert Juan-Arinyo.

LSI-94-21-R "Clausal proof nets and discontinuity", Glyn Morrill.


LSI-94-23-R "On the sparse set conjecture for sets with low density", Harry Buhrman and Montserrat Hermo.

Copies of reports can be ordered from:

Nuria Sánchez
Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Pau Gargallo, 5
08028 Barcelona, Spain
secrelsi@lsi.upc.es