# ON A SUBCLASS OF PSEUDOPOLYNOMIAL PROBLEMS [1]

BURKHARD MONIEN
UNIVERSITÄT PADERBORN
PADERBORN, WEST - GERMANY

ABSTRACT

A subclass of the class of all pseudopolynomial problems is defined as a family of sets acceptable by some automaton operating with simultaneous time and space bounds. That the class is large enough can be seen in that it contains many (if not all) of the pseudopolynomial problems described in the literature. We study structure preserving reductions within this class and give intuitive reasons (borrowed from our knowledge about space bounded automata) that there exist at least four well known problems which are pairwise not equivalent under these reductions.

## 1. INTRODUCTION

In this paper we consider problems which are $\mathbb{NP}$-complete but whose complexity depends polynomially on some number defined by the input. We assume that the reader is familiar with the notions $\mathbb{P}$, and $\mathbb{NP}$-complete. Let our reductions "$\leq$" be those defined by deterministic log-space bounded Turing machines.

Though all $\mathbb{NP}$-complete problems have the same worst case behaviour up to polynomial transformations, there exist $\mathbb{NP}$-complete problems which behave numerically well in most applications.

We consider as an example the subset sum problem

$$\text{SUB} = \{a_1 ¢ \ldots ¢ a_n ¢ b \mid n,b,a_i \in \mathbb{N},\ 1 \leq i \leq n, \text{ and } \exists\, I \subset \{1,\ldots,n\} : \sum_{i\in I} a_i = b\}$$

where we assume that the numbers $a_i$, $b$, $1 \leq i \leq n$, are encoded by their binary notation. SUB is $\mathbb{NP}$-complete ([6]), but on the other hand it is well known that SUB is solved also by a deterministic algorithm working within the time-bound $O(n \cdot b)$. Note that this does not imply that SUB belongs to $\mathbb{P}$, since $b$ grows exponentially with the length of its binary notation.

There have been two approaches to formalize this behaviour: (1) M.R. Garey and D.S. Johnson ([5]) introduce a function Max: {correct encodings} $\to \mathbb{N}$ that associates to the encoding of a problem the largest number occuring in this encoding. They call a problem pseudopolynomial if there exists an algorithm which works for any input $w$ with the time bound $O((|w| + \text{Max}(w))^q)$ for some $q \in \mathbb{N}$.

(2) Paz, Moran ([10]) and Ausiello et al. ([1]) consider optimization problems. They define the notions "simple" and "p-simple". We will not give their definitions here. If a p-simple optimization problem is replaced by an encoding as a language (in the usual way $f(x) = \text{Max}$ is replaced by $\exists x : f(x) \geq D$), then this language is pseudopolynomial with the additional property that the corresponding algorithm is also polynomial in $D$.

We use in this paper a formalization which generalizes the notion "pseudopolynomial" in the following way: We associate to a problem a function $g$: {correct encodings} $\to \mathbb{N}$ and we call the problem pseudopolynomial if it can be solved for any input $w$ with the time bound $O((|w| + g(w))^q)$ for some $q \in \mathbb{N}$. Of course, now a problem is pseudopolynomial only in connection with this function $g$. We get the old notion of "pseudopolynomial" if we take $g = \text{Max}$. (Actually our definition will be still more general by allowing relations instead of functions.)

In order to state this definition formally we consider sets $R \subset X^* \times X^*$ for some alphabet $X$.

Let $R \subset X^* \times X^*$ be some set and let $f: \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be some function. We say that $R$ is accepted by a Turing machine (this machine may be deterministic or nondeterministic) within the time bound or space bound, respectively, $f(n,m)$, if $M$ accepts $(u,v) \in X^* \times X^*$ iff $(u,v) \in R$ and if for any $(u,v) \in R$ there exists an accepting computation which needs not more than $f(|u|, |v|)$ steps (or cells, respectively).

<u>Definition</u>: $R \subset X^* \times X^*$ is called pseudopolynomial iff there exists some polynomial $p$, some $d \in \mathbb{N}$ and some deterministic Turing machine accepting $R$ within the time-bound $p(n+m) \cdot d^m$.

Note that the complexity of accepting a pair $(u,v)$ grows polynomial with the number which is encoded by $v$. We denote by PP the family of all pseudopolynomial sets.

We define now reductions between sets of pairs in such a way that the number given by the second component grows at most polynomially (that means that the length of the second component grows at most linearly).

<u>Definition</u>: For $R_1, R_2 \in$ PP we say $R_1 \leq_\pi R_2$ if there exist functions $f_1, f_2 \in \text{DSPACE}(\log n)$ such that

(1) $(u,v) \in R_1 \Leftrightarrow (f_1(u,v), f_2(u,v)) \in R_2 \qquad \forall\, u,v \in X^*$

(2) $|f_1(u,v)| \geq |u|$ and $\exists\, c \in \mathbb{N} : |f_2(u,v)| \leq c \cdot |v|$

We use this more general notion of "pseudopolynomial" since:

- under this definition the class PP is closed under $\leq_\pi$ reductions and this allows us to speak about "complete" problems
- there were very good practical reasons to define "pseudopolynomial" in terms of maximum numbers as in [5], since many scheduling and knapsack-like problems are pseudopolynomial in this sense (and in fact all the applied problems we consider in the next section are also pseudopolynomial under the old definition). On the other hand, if we are interested in the structure of $\mathbb{NP}$-complete problems, we may ask whether there are other structural properties than just the maximal number which make a problem behave pseudopolynomially. It is shown in [9] that for graph theory problems the bandwidth of the graph plays the same role as the length of the maximal number does for scheduling problems, i.e. for many graph theory problems the set

[1] Some of this work was done while the author visited the department of Mathematics, University of California at Santa Barbara, where it was supported in part by the National Science Foundation under grant MCS 77 - 1136o.

$\{(u,v) \mid u$ is an encoding of a graph G with the given property and $v = 0^{f(u)}$, where $f(u)$ = bandwith of the graph encoded by $u\}$ is pseudopolynomial in our sense.

– we will show that there exists a problem which is complete for RPP, the class of restricted pseudopolynomial problems ( defined below), and which is pseudopolynomial in the sense of [ 5 ]. This means that RPP is just the $\leq_\pi$-closure of a problem which is pseudopolynomial in the old sense.

Definition: $R \subset X^* \times X^*$ is called a restricted pseudopolynomial problem (and the class of all such problems is denoted by RPP) iff there exists some nondeterministic Turing machine accepting R with the space bound $\max\{\log n, m\}$ and the simultaneous time bound $(n+m)^q$ for some $q \in \mathbb{N}$.

This definition implies that for any $R \in$ RPP the language $L_R = \{u \not{c}\, v \mid (u,v) \in R\}$ belongs to NP . Furthermore the nondeterministic Turing machine accepting R within the space bound $\max\{\log n, m\}$ can be simulated by some deterministic Turing machine accepting R within the time bound $n^q \cdot d^m$ for some $q, d \in \mathbb{N}$.

This implies that RPP $\subset$ PP. Furthermore we note that the deterministic algorithm which simulates the nondeterministic tape bounded Turing machine is a so called "dynamic programming" algorithm. We will see in the next section that RPP contains many of the pseudopolynomial problems studied in the literature. The algorithms which are given in the literature for solving these problems are also "dynamic programming" algorithms reflecting just the behaviour of the corresponding nondeterministic space bounded Turing machine. We feel this observation simplifies the search for pseudopolynomial-time algorithms since it is generally easier to define a space bounded automaton than to construct the corresponding "dynamic programming" algorithm.

It is clear that there exists a close relationship between RPP and the classes of languages defined by nondeterministic Turing machines operating with sublinear space bounds and polynomial time bounds simultaneously. Let us denote by NPTIME SPACE(f) the class of all languages accepted by some nondeterministic Turing machine within polynomial time and simultaneous space bound f. Let us further associate to each $R \in$ PP and to each function $f: \mathbb{N} \to \mathbb{N}$ the language $L_R(f) = \{u \not{c}\, v \mid (u,v) \in R$ and $|v| \leq f(|u|)\}$. Then the following theorem holds.

Theorem 1: Let f be any monotonic increasing function which is computable by a deterministic log-space bounded Turing machine and which fulfills $f(n) \geq \log n \quad \forall n \in \mathbb{N}$.

(1) PP and RPP are closed under $\leq_\pi$ reductions

(2) $R_1, R_2 \in$ PP, $R_1 \leq_\pi R_2 \Rightarrow L_{R_1}(f) \leq L_{R_2}(f)$

(3) $R \in$ RPP $\Rightarrow L_R(f) \in$ NPTIMESPACE(f)

(4) R is complete for RPP with respect to $\leq_\pi$
$\Rightarrow L_R(f)$ is complete for $\bigcup_d$ NPTIMESPACE$(f(n^d))$ with respect to $\leq$.

Proof: (1) We have to show that $R_1 \leq_\pi R_2$ and $R_2 \in$ PP(RPP) implies $R_1 \in$ PP(RPP). Suppose $R_1 \leq_\pi R_2$. Then there exist $g_1, g_2 \in$ DSPACE(log n) such that $|g_2(u,v)| \leq c \cdot |v|$ for some $c \in \mathbb{N}$ and $(u,v) \in R_1 \Leftrightarrow (g_1(u,v), g_2(u,v)) \in R_2$. Therefore if $R_2 \in$ PP then there exists a deterministic Turing machine accepting $R_1$ with the time bound



$(|g_1(u,v)|+|g_2(u,v)|)^q \cdot d^{|g_2(u,v)|} \leq (|u|+|v|)^{\tilde{q}} \cdot \tilde{d}^{|v|}$ for some $q, d, \tilde{q}, \tilde{d} \in \mathbb{N}$. If $R_2 \in$ RPP then there exists a nondeterministic Turing machine accepting $R_1$ with the time bound $(|g_1(u,v)|+|g_2(u,v)|)^q \leq (|u|+|v|)^{\tilde{q}}$ and the space bound $\max\{\log |g_1(u,v)|, |g_2(u,v)|\} \leq \tilde{c} \cdot \max\{\log|u|, |v|\}$. Therefore $R_1 \in$ PP ($R_1 \in$ RPP, respectively).

(2) Suppose $R_1, R_2 \in$ PP and $R_1 \leq_\pi R_2$. Then there exist $g_1, g_2 \in$ DSPACE(log n) such that $|g_1(u,v)| \geq |u|$, $|g_2(u,v)| \leq c \cdot |v|$ for some $c \in \mathbb{N}$ and $(u,v) \in R_1 \Leftrightarrow (g_1(u,v), g_2(u,v)) \in R_2$. If $R_2 = X^* \times X^*$ then also $R_1 = X^* \times X^*$ and obviously $R_1 = R_2$ implies $L_{R_1}(f) = L_{R_2}(f)$. Now suppose $R_2 \neq X^* \times X^*$ and take $(u_o, v_o) \in X^* \times X^* - R_2$. Define g by $g(u \not{c}\, v) = (u_o, v_o)$ if $|v| > f(|u|)$ and $g(u \not{c}\, v) = g_1(u,v) \not{c}\, g_2(u,v)$ otherwise. Then $g \in$ DSPACE(log n) and $u \not{c}\, v \in L_{R_1}(f) \Leftrightarrow (u,v) \in R_1$ and $|v| \leq f(|u|) \Rightarrow (g_1(u,v), g_2(u,v)) \in R_2$ and $|g_2(u,v)| \leq c \cdot |v| \leq c \cdot f(|u|) \leq c \cdot f(g_1(u,v)) \Leftrightarrow g(u \not{c}\, v) \in L_{R_2}(f)$. On the other hand, if $g(u \not{c}\, v) \in L_{R_2}(f)$ then $(g_1(u,v), g_2(u,v)) \in R_2$ and (because of the definition of g) $|v| \leq f(|u|)$. This implies $u \not{c}\, v \in L_{R_1}(f)$. So we have shown that $u \not{c}\, v \in L_{R_1}(f) \Leftrightarrow g(u \not{c}\, v) \in L_{R_2}(f)$.

(3) Let M be a nondeterministic Turing machine accepting R with the time bound $(n+m)^q$ for some $q \in \mathbb{N}$ and with the space bound $\max\{\log n, m\}$. We define a nondeterministic Turing machine M', accepting $L_R(f)$, in the following way: (i) M' tests whether its input has the form $u \not{c}\, v$ with $u, v \in X^*$ and $|v| \leq f(|u|)$; (ii) M' simulates the behaviour of M on (u,v). Clearly M' operates in polynomial time and with the space bound f.

(4) Let L be a language which is complete for $\bigcup_d$ NPTIMESPACE $(f(n^d))$. We can assume that $L \in$ NPTIMESPACE(f). Set $\tilde{R} = \{(u, 0^{f(|u|)}) \mid u \in L\}$.
We want to show that $\tilde{R} \in$ RPP. Let M be a nondeterministic Turing machine accepting L within polynomial time and the space bound f. We define a nondeterministic Turing machine M' accepting $\tilde{R}$ in the following way: (i) M' checks whether the input has the form (u,v) with $u \in X^*$ and $v = 0^{f(|u|)}$, (ii) M' simulates the behaviour of M on the string u. Obviously M operates within polynomial time and the space bound $|v|$. Therefore $\tilde{R} \in$ RPP and since R is complete for RPP we get $\tilde{R} \leq_\pi R$. Since f is computable by some deterministic log-space bounded Turing machine, $L \leq L_{\tilde{R}}(f) = \{u \not{c}\, 0^{f(|u|)} \mid u \in L\}$ and because of (2) this implies $L \leq L_{\tilde{R}}(f) \leq L_R(f)$. Because of (3) $L_R(f) \in$ NPTIMESPACE(f) and therefore $L_R(f)$ is complete for this class. □

## 2. REDUCTIONS BETWEEN CONCRETE PROBLEMS

We consider in this section the following problems whose language-encodings are all known to be complete for NP.

(1) subset sum

RSUB = $\{(a_1 \not{c} \ldots \not{c}\, a_n, b) \mid a_1 \not{c} \ldots \not{c}\, a_n \not{c}\, b \in$ SUB$\}$

(2) partition into k subsets

$\mathrm{RPAR}(k) = \{(a_1 ¢ \dots ¢ a_n, \sum_{i=1}^{n} a_i) \mid \exists I_1, \dots, I_k \subset \{1,\dots,n\}:$
$\bigcup_{\nu=1}^{k} I_\nu = \{1,\dots,n\},\ I_\nu \cap I_\mu = \emptyset$ for $\nu \neq \mu$ and
$\sum_{i\in I_\nu} a_i = \sum_{i \in I_\mu} a_i$ for all $\nu, \mu \in \{1,\dots,k\}\}$

(3) multi-processor scheduling on k processors

$\mathrm{RMPS}(k) = \{(a_1 ¢ \dots ¢ a_n, D) \mid \exists I_1, \dots, I_k \subset \{1,\dots,n\}:$
$\bigcup_{\nu=1}^{k} I_\nu = \{1,\dots,n\}$ and $\sum_{i\in I_\nu} a_i \le D$ for all $\nu = 1, \dots, k\}$

(4) sequencing to minimize tardy task weight

$\mathrm{RTTW} = \{(t_1 ¢ w_1 ¢ d_1 ¢ \dots ¢ t_n ¢ w_n ¢ d_n ¢ W, \max(W, \sum_{i=1}^{n} t_i)) \mid$
$\exists$ permutation $\sigma: \{1,\dots,n\} \to \{1,\dots,n\}$ such that
$\sum_{\nu\in I} w_\nu \le W$ where $I = \{\nu \mid \sum_{i=1}^{\nu} t_{\sigma(i)} > d_{\sigma(\nu)}\}\}$

(5) scheduling to minimize weighted mean flow time on k processors

$\mathrm{RWMFT}(k) = \{t_1 ¢ w_1 ¢ \dots ¢ t_n ¢ w_n, W) \mid \exists I_1,\dots,I_k \subset \{1,\dots,n\}$
and permutations $\sigma_\nu: \{1,\dots, |I_\nu|\} \to I_\nu$, $1 \le \nu \le k$, such that
$\bigcup_{\nu=1}^{k} I_\nu = \{1,\dots,n\}$ and $\sum_{\nu=1}^{k} \sum_{i=1}^{|I_\nu|} w_{\sigma_\nu(i)} \sum_{\sigma_\nu(j)\le\sigma_\nu(i)} t_{\sigma_\nu(j)} \le W\}$

(6) selecting numbers from blocks of length k

$\mathrm{RSEL}(k) = \{(a_1 ¢ \dots ¢ a_{k\cdot n}, b) \mid \exists I \subset \{1,\dots,n\}: \sum_{i\in I} a_i = b$
and $|I \cap \{a_{i+1},\dots, a_{i+k}\}| = 1 \quad \forall\, i=0, \dots, n-1\}$

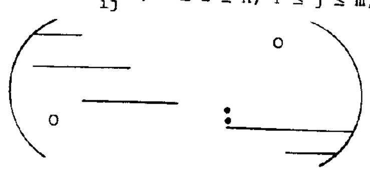
selecting numbers from blocks of arbitrary length

$\mathrm{RSEL} = \{(a_{11} ¢ \dots ¢ a_{1l_1} \# \dots \# a_{n1} ¢ \dots ¢ a_{nl_n}, b) \mid$
$\exists j_1,\dots,j_n : 1 \le j_i \le l_i \quad \forall i$ and $\sum_{i=1}^{n} a_{ij_i} = b\}$

(7) solving a linear equation with lower bounds on the subsums

$\mathrm{RLBS} = \{(a_1 ¢ d_1 ¢ \dots ¢ a_n ¢ d_n, b) \mid \exists x_1, \dots, x_n \in \{0,1\}:$
$\sum_{i=1}^{n} a_i x_i = b$ and $\sum_{i=1}^{j} a_i x_i \ge d_j \quad \forall j = 1,\dots,n\}$

(8) solving a system of linear equations where the associated matrix $A = (a_{ij})$, $1 \le i \le n$, $1 \le j \le m$, has the form

A = 

i.e. there exist $d_i \in \mathbb{N}$, $1 \le i \le n+2$
$d_1 \le d_2 \le \dots \le d_{n+2}, d_{n+1} = d_{n+2} = n$
such that for all $i=1,\dots,n$: $a_{ij} = 0$
for $j < d_i$ and for $j \ge d_{i+2}$

$\mathrm{RLSE} = \{(a_{11} ¢ a_{12} ¢ \dots ¢ a_{nm} ¢ b_1 ¢ \dots ¢ b_m, \max_i b_i) \mid A = (a_{ij})$
fulfills the above condition, $a_{ij} \in \mathbb{N}$, and $\exists x_1, \dots, x_m \in \{0,1\}$
such that $\sum_{j=1}^{m} a_{ij} x_j = b_i \quad \forall i = 1,\dots,n\}$

(9) looking for a path with nonnegative weights



$\mathrm{RGAP} = \{(E ¢ z_1 ¢ \dots ¢ z_n, \max |z_i|) \mid z_i \in \mathbb{Z} \quad \forall i = 1,\dots,n:$
$E \subset \{1,\dots,n\} \times \{1,\dots,n\}$ and $(\{1,\dots,n\}, E)$ forms an acylic graph;
there exist $r \in \mathbb{N}$ and $k_1, \dots, k_r \in \{1,\dots,n\}$ such
that $k_1 = 1$, $k_r = n$, $(k_i, k_{i+1}) \in E \quad \forall i = 1,\dots,r-1$ and
$\sum_{\nu=1}^{i} z_{k_\nu} \ge 0 \quad \forall i = 1,\dots,r-1$ and $\sum_{\nu=1}^{r} z_{k_\nu} = 0\}$

(10) solving a set of quadratic diophantine equations

$\mathrm{RQDE} = \{(a_1 ¢ \dots ¢ a_n ¢ b, d) \mid a_i, b, d \in \mathbb{N};\ \forall i\ \exists x_i, y_i \in \mathbb{N}: a_i x_i^2 + b y_i = d\}$

We show first that all these problems belong to RPP. Because of theorem 1, (1) and the reductions which will be given in theorem 3 we have only to show:

<u>Theorem 2:</u> RLBS, RLSE, RWMFT(k) $\in$ RPP

<u>Proof:</u> (1) RLBS is accepted by a nondeterministic Turing machine M which scans with its input head the string $a_1 ¢ d_1 ¢ \dots ¢ a_n ¢ d_n$ and which stores on its work tape the sum S of all the $a_j$ which have been chosen up to this point. When M reaches the number $a_i$ it decides whether to set $x_i = 0$ or $x_i = 1$ and it sets $S := S + a_i$ if it has decided to set $x_i = 1$. Afterwards it checks whether $d_i \le S \le b$ holds. Having scanned the whole encoding it decides whether $S = b$ holds. Obviously $S \le b$ holds during the whole computation. Therefore RLBS $\in$ RPP.

(2) In order to solve RLSE a Turing machine has to guess $x_1, \dots, x_m \in \{0,1\}$. Suppose now our machine M is in a configuration where it has to decide whether to set $x_k = 0$ or $x_k = 1$. Instead of storing $x_1,\dots,x_{k-1}$ (which would need too much space) it stores all the information it needs about the sums $\sum_{j=1}^{k-1} a_{ij} x_j$, $1 \le i \le n$. Let q be determined by $d_q \le k < d_{q+1}$. Then because of the structure of A $\sum_{j=1}^{k-1} a_{ij} x_j = 0$ for $i \ge d_{q+1}$ (and therefore these sums need not be stored) and $\sum_{j=1}^{k-1} a_{ij} x_j = \sum_{j=1}^{m} a_{ij} x_j$ for $i \le d_{q-2}$ and therefore A can only have a solution if $\sum_{j=1}^{k-1} a_{ij} x_j = b_i$ for $1 \le i \le d_{q-2}$. M has checked this before it reached our configuration and therefore M has only to store the numbers $y = \sum_{j=1}^{k-1} a_{d_{q-1}j} x_j$, $Z = \sum_{j=1}^{k-1} a_{d_q j} x_j$.

Having decided whether to set $x_k = 0$ or $x_k = 1$ M leaves these numbers unchanged or it changes them to $y := y + a_{d_{q-1}k}$, $Z := Z + a_{d_q k}$. M stops if one of these numbers is larger than $\max_i b_i$. It is obvious that M works with the space bound $\max_i |b_i|$ and therefore RLSE $\in$ RPP.

(3) It is wellknown (see [ 3 ]) that if an instance of RWMFT(k) has a solution then there exists also a solution such that $t_{\sigma_\nu(i)}/w_{\sigma_\nu(i)} \le t_{\sigma_\nu(i+1)}/w_{\sigma_\nu(i+1)}$ holds for all $1 \le \nu \le k$ and for all $1 \le i \le |I_\nu| - 1$. Set $R_1(k) = \{(t_1 ¢ w_1 ¢ \dots ¢ t_n ¢ w_n, W) \mid (t_1 ¢ w_1 ¢ \dots ¢ t_n ¢ w_n, W) \in \mathrm{RWMFT}(k)$ and $t_i/w_i \le t_{i+1}/w_{i+1} \quad \forall\, 1 \le i \le n-1\}$. It is obvious that $\mathrm{RWMTF}(k) \le_\pi R_1(k)$.

We define a nondeterministic Turing machine M accepting $R_1(k)$. Its storage tape is divided into k+1 tracks and when it has to decide on which of the k processors the j-th task has to be performed, then it has already put all the tasks $1,\dots,j-1$ into one of the set $I_\nu$, $1 \le \nu \le k$, and it stores on its k+1 tracks the numbers $y_\nu = \sum_{i\in I_\nu,\, i\le j-1} t_i$, $1 \le \nu \le k$ and $Z = \sum_{\nu=1}^{k} \sum_{i\in I_\nu, i\le j-1} w_i \sum_{p\in I_\nu, p\le i} t_p$. When it decides that the j-th task has to be performed on processor q, $1 \le q \le k$, then it changes these numbers into $y_q := y_q + t_j$, $Z := Z + w_j\, y_q$. At the end M has to decide whether $Z \le W$ holds and it is clear that M can be constructed in such a way that all the numbers stored on its k+1 tracks are bounded by W during its whole computation. Therefore $R_1(k)$ belongs to RPP and because of theorem 1,(1) RWMFT(k) $\le_\pi R_1(k)$ implies that RWMFT(k) belongs to RPP. $\square$

In the next theorem we state those reductions between these problems which we are able to find. In the next section we will give some intuitive reasons that at least four of these problems are not reducible to each other. This indicates that $\le_\pi$ reductions define a rich structure among the problems belonging to our class RPP.

<u>Theorem 3:</u> RLSE is complete for RPP and there are reductions between the problems (1), ..., (1o) as shown in diagramm 1.

<u>Proof:</u> (1) It was shown in [7] that RLSE is complete for RPP (though a different notation was used in [7]).

(2) It is clear that $R\,\alpha(k) \le R\,\alpha(k+1)$ for $\alpha$ = PAR, MPS, WMFT, SEL and for all $k \in \mathbb{N}$ and that RSEL(k) $\le$ RSEL for all $k \in \mathbb{N}$. Furthermore RLBS, RQDE, RWMFT(k) $\le$ RLSE $\forall k\in\mathbb{N}$, since all these sets belong to RPP and RLSE is complete for RPP.

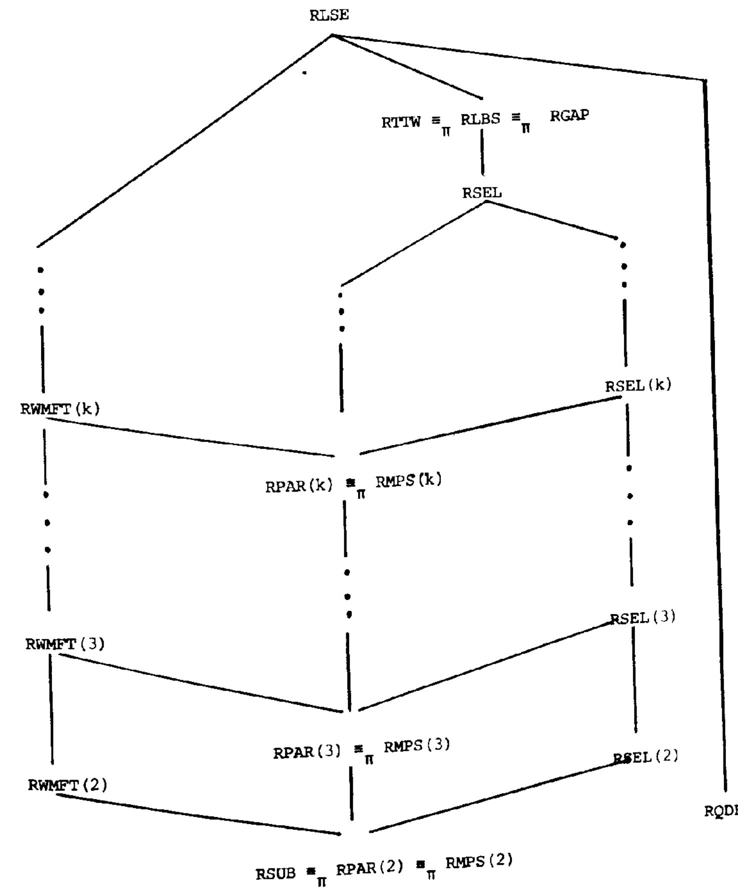
(3) We want to show RSUB $\equiv_\pi$ RPAR(2). RPAR(2) $\le$ RSUB holds since $(a_1 ¢ \dots ¢ a_n, \max_i a_i) \in$ RPAR(2) iff $\sum_{i=1}^{n} a_i \equiv 0 \bmod 2$ and $(a_1 ¢ \dots ¢ a_n, \frac{1}{2}\sum_{i=1}^{n} a_i) \in$ RSUB. In order to show RSUB $\le$ RPAR(2) we have to consider two cases. Suppose an instance $(a_i ¢ \dots ¢ a_n, b)$ is given and $\sum_{i=1}^{n} a_i \ge b$. Set $A = \sum_{i=1}^{n} a_i$. If $A \le 2b$ then $(a_1 ¢ \dots ¢ a_n, b) \in$ RSUB iff $(a_1 ¢ \dots ¢ a_n ¢ 2b - A, 2b) \in$ RPAR(2). If $A \ge 2b$ then $(a_1 ¢ \dots ¢ a_n, b) \in$ RSUB iff $(a_1 ¢ \dots ¢ a_n ¢ A-2b, 2A-2b) \in$ RPAR(2).

(4) We have to show RPAR(k) $\equiv_\pi$ RMPS(k) $\forall k\in\mathbb{N}$. RPAR(k) $\le$ RMPS(k) since $(a_1 ¢ \dots ¢ a_n, \max_i a_i) \in$ RPAR(k) iff $\sum_{i=1}^{n} a_i \equiv 0 \bmod k$ and $(a_1 ¢ \dots ¢ a_n, \frac{1}{k}\sum_{i=1}^{n} a_i) \in$ RMPS(k). In order to show RMPS(k) $\le$ RPAR(k) let $(a_1 ¢ \dots ¢ a_n, D)$ be an instance of the multiprocessor scheduling. Set $E = k\cdot D - \sum_{i=1}^{n} a_i$. Let $p, q \in \mathbb{N}$ be defined by: p is the maximal number such that $k(2^{p+1} - 1) \le E$ and q is the maximal number such that $q\cdot 2^{p+1} \le E - k\cdot(2^{p+1} - 1)$. (Note that $p \le \log_2 E$ and that $q < k$.) Then $(a_1 ¢ \dots ¢ a_n, D) \in$ RMPS(k) iff $(a_1 ¢ \dots ¢ a_n ¢ b_1 ¢ \dots ¢ b_{k\cdot(p+1)} ¢ c_1 ¢ \dots ¢ c_q ¢ d, D) \in$ RPAR(k), where $b_{i\cdot k+\nu} = 2^i$ for $0 \le i \le p$, $1 \le \nu \le k$ and $c_j = 2^{p+1}$ for $1 \le j \le q$ and $d = E - k\cdot(2^{p+1} - 1) - q\cdot 2^{p+1}$.

(5) RPAR(k) $\le_\pi$ RSEL(k) since $(a_1 ¢ \dots ¢ a_n, \max_i a_i) \in$ RPAR(k) iff $\sum_{i=1}^{n} a_i = 0$ mod k and $(b_1 ¢ \dots ¢ b_{k\cdot n}, D\cdot \sum_{i=0}^{k-1} A^i) \in$ SEL(k), where $A = \sum_{i=1}^{n} a_i$ and $D = A/k$ and $b_{k\cdot i+1} = a_{i+1} A^{j-1}$ for $0 \le i \le n-1$, $1 \le j \le k$.

(6) RPAR(k) $\le_\pi$ RWMFT(k) was shown in [11]. S.K. Sahni showed that $(a_1 ¢ \dots ¢ a_n, \max_i a_i) \in$ RPAR(2) iff $\sum_{i=1}^{n} a_i = 0 \bmod 2$ and $(a_1 ¢ a_1 ¢ \dots$

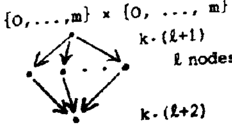





<u>Diagram 1</u>

$\dots ¢ a_n ¢ a_n, \frac{1}{2}\sum_{i=1}^{n} a_i^2 + \frac{1}{4}(\sum_{i=1}^{n} a_i)^2) \in$ RWMFT(2).

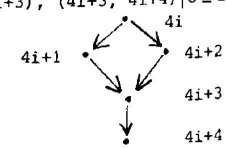
(7) In order to show RSEL $\le_\pi$ RGAP let us first mention that RSEL $\le_\pi R_1$, where $R_1$ consists of all those selection problems where all blocks have the same length. (RSEL $\le_\pi R_1$ holds since we can fill a block up to a given length with one of its elements). Now we have to show $R_1 \le_\pi$ RGAP and this is true since $(a_{11} ¢ \dots ¢ a_{1\ell} ¢ \dots ¢ a_{n1} ¢ \dots ¢ a_{n\ell}, b) \in R_1$ iff $(E ¢ 0 ¢ a_{11} ¢ \dots ¢ a_{1\ell} ¢ 0 ¢ \dots ¢ 0 ¢ a_{n1} ¢ \dots ¢ a_{n\ell} ¢ -b, \dots) \in$ RGAP, where $E \subset \{0,\dots,m\} \times \{0,\dots,m\}$, $m = n\cdot(\ell+1)$, is defined by $(i,j) \in E \Leftrightarrow$ $1 \le j - i \le \ell$ and $[i = k\cdot(\ell+1)$, $0 \le k < n$ or $j = k\cdot(\ell+1)$, $1 \le k \le n]$.



(8) Now it remains to prove that RTTW $\equiv_\pi$ RLBS $\equiv_\pi$ RGAP. We will show that RGAP $\leq_\pi$ RLBS $\leq_\pi$ RTTW $\leq_\pi$ RGAP holds. In [7] the author showed that GAP $\leq L_{RLBS}(\log n)$ and with the same method it is shown in [8] that RGAP $\leq_\pi$ RLBS holds.

In order to show RLBS $\leq_\pi$ RTTW let us consider more closely an instance of RTTW. If we have given $(t_1 ¢ w_1 ¢ d_1 ¢ \ldots ¢ t_n ¢ w_n ¢ d_n ¢ W, \max(W, \sum_{i=1}^{n} t_i))$, then we can assume that all jobs that cannot be processed by their deadline are processed at the end of the schedule in an arbitrary order. Therefore RTTW $\equiv R_1$, where $R_1 = \{(\ldots,\ldots) \mid \exists I \subset \{1,\ldots,n\} : \sum_{\nu \in I} w_\nu \leq W$ and $\sum_{\overline{\nu \in I, \nu < i}} t_\nu \leq d_i$ for all $i = 1,\ldots,n\}$. Using standard techniques we can replace the condition $\sum_{\nu \in I} w_\nu \leq W$ by $\sum_{\nu \in I} w_\nu = W$ and replacing finally I by $\{1,\ldots,n\} - I$ we get RTTW $\equiv R_2$, where $R_2 = \{(t_1 ¢ w_1 ¢ d_1 ¢ \ldots ¢ t_n ¢ w_n ¢ d_n ¢ W, \max(W, \sum_{i=1}^{n} t_i)) \mid \exists I \subset \{1,\ldots,n\} : \sum_{\nu \in I} w_\nu = W$ and $\sum_{\overline{\nu \in I, \nu < i}} t_\nu > d_i \ \forall i=1,\ldots,n\}$. It is obvious that RLBS $\leq_\pi R_2$ since RLBS is just the special case of $R_2$ where $w_\nu = t_\nu \ \forall \nu = 1, \ldots, n$.

We now have to show RTTW $\leq$ RGAP. By using standard techniques it is not difficult to verify that $R_2 \equiv_\pi R_3$ where $R_3 = \{(t_1 ¢ w_1 ¢ d_1 ¢ \ldots ¢ t_n ¢ w_n ¢ d_n ¢ W ¢ T, \max(W,T)) \mid \exists I \subset \{1,\ldots,n\} : \sum_{\nu \in I} w_\nu = W, \sum_{\nu \in I} t_\nu = T$ and $\sum_{\overline{\nu \in I, \nu < i}} t_\nu > d_i \ \forall i = 1,\ldots,n\}$. $R_3 \leq$ RGAP, since $(t_1 ¢ \ldots ¢ d_n ¢ W ¢ T, \ldots) \in R_3$ iff $(E ¢ a_0 ¢ a_1 ¢ \ldots ¢ a_{4n}, \ldots) \in$ RGAP where $E = \{(4i, 4i+1), (4i, 4i+2), (4i+1, 4i+3), (4i+2, 4i+3), (4i+3, 4i+4) \mid 0 \leq i \leq n-1\}$ and $a_{4i+1} = 0$, $a_{4i+2} = t_{i+1} \cdot \widetilde{W} + w_{i+1}$ for all $0 \leq i \leq n-1$ (where $\widetilde{W} = \sum_{i=1}^{n} w_i$) and $a_{4i+3} = -(d_{i+1} + 1)\widetilde{W}$, $a_{4i+4} = (d_{i+1} + 1)\widetilde{W}$ for all $0 \leq i \leq n-2$ and $a_0 = a_{n+4} = 0$ and $a_{n+3} = -W \cdot \widetilde{W} - T$.




□

We don't intend to give an exhaustive list of all the problems belonging to RPP. Let us mention as further examples only the following problems: "solving a system of k linear equations" (equivalent to RSUB), "sequencing with $k_1$ release times and $k_2$ deadlines" (see [5], reducible to RSEL) and "sequencing with set-up times and k deadlines" (see [5], reducible to RSEL). We can construct additional problems which are complete for RPP by making our problems slightly more difficult, e.g. "finding a solution $x_i \in \{0,1\}$ of $\sum_{i=1}^{n} a_i x_i = b$, $\sum_{\nu=1}^{i} c_{j\nu} \geq d_i$ for $j = 1,2,3$ and $1 \leq i \leq n$" is complete for RPP. In [9] a great number of problems are shown to be complete for RPP just by taking various graph theory problems where the bandwidth of the graph is used as the structural information, e.g. the problem of finding a 3-colouring for a graph is complete for RPP (remember that we consider as inputs pairs $(G, 0^m)$ where m is the bandwidth of G).

## 3. CONNECTIONS TO SPACE BOUNDED COMPUTATIONS

In this section we will give intuitive reasons to support the conjecture that some of our problems are not reducible to each other. At the moment no method is known to prove that such reductions do not exist. It is an open question whether or not NP is equal to DSPACE(log n). We prove the following lemma:



Lemma 1: Let a be any symbol. If NP = DSPACE(log n) then $R \leq_\pi \{(v,\varepsilon) \mid v \in \{a\}^*\}$ holds for any $R \in$ RPP.

Proof: Suppose NP=DSPACE(log n). Then for any $R \in$ RPP the set $L_R = \{u ¢ v \mid (u, v) \in R\}$ belongs to DSPACE(log n) and $R \leq_\pi \widetilde{R}$, where $\widetilde{R} = \{(u ¢ v, \varepsilon) \mid (u,v) \in R\}$. There exists a function f which is computable by a deterministic log space bounded Turing machine such that $|f(w)| \geq |w|$ and $w \in L_R$ iff $f(w) \in \{a\}^*$. This implies $\widetilde{R} \leq_\pi \{(v,\varepsilon) \mid v \in \{a\}^*\}$. □

The next lemma gives the basis for our intuitive reasons.

Lemma 2: (1) $L_{RQDE}(\log n) \in$ DSPACE(log n)

(2) There exists a language $L_1$ which is acceptable by some nondeterministic one-way reversal-bounded counter automaton (for a definition see [2], [8]) such that $L_{RSEL}(\log n) \equiv L_1$.

(3) $L_{RTTW}(\log n)$ is complete for NSPACE(log n).

Proof: (1) In order to solve $a_i x^2 + by = c$ we have only to consider x,y with x,y $\leq$ c. This can be done by some deterministic Turing machine with the space bound $|c|$.

(2) Set $L_1 = \{0^{a_{11}} 1 0^{a_{12}} 1 \ldots 1 0^{a_{1\ell_1}} 11 \ldots 11\, 0^{a_{n1}} 1 \ldots 1 0^{a_{n\ell_n}} 111\, 0^b \mid$
$\exists j_1,\ldots,j_n : 1 \leq j_i \leq \ell_i$ such that $\sum_{i=1}^{n} a_{ij_i} = b\}$. Note that the mapping $c_1$, associa-
ting to the binary encoding of a number its unary encoding, is computable by a deter-
ministic Turing machine with a linear space bound and that its inverse mapping $c_1^{-1}$
is computable by a deterministic Turing machine with the space bound log n. There-
fore $R_1 \equiv L_{RSEL}(\log n)$ holds. Furthermore we can easily construct a nondeterministic one-way counter automaton M which changes the direction of its counter exactly once and which accepts $L_1$. M scans with its input head the input string from left to right and it takes from each block i exactly one number $a_{ir_i}$ and adds it to its counter (i.e. while scanning $0^{a_{ir_i}}$ it increases the counter by 1 in each step). After it has reached 111 it starts to decrease the counter and checks whether the number stored by the counter is equal to b. □

(3) It was shown in [7] that $L_{RLBS}(\log n)$ is complete for NSPACE(log n) and we showed in theorem 2 that RLBS $\equiv_\pi$ RTTW holds.

We are now ready to state as conjectures that the following reductions do not exist:

Conjecture (1) RSEL is not reducible to RQDE

Conjecture (2) RTTW is not reducible to RSEL

Conjecture (3) RSLE is not reducible to RTTW

Note that because of the reductions given in theorem 1 "RSEL is not reducible to RQDE" implies that also RTTW and RLSE are not reducible to RQDE and that "RTTW is not reducible to RSEL" implies that also RLSE is not reducible to RSEL and that RTTW is not reducible to RMPS(k) for any $k \in \mathbb{N}$. Quite a

Reasons: (1) Because of lemma 2 $L_{RQDE}(\log n)$ belongs to DSPACE(log n).

lot of work has been done to show $L_{RSUB}(\log n) \in DSPACE(\log^k n)$ for some $k < 2$. Since these efforts have failed we conjecture that $L_{RSEL}(\log n)$ does not belong to DSPACE $(\log n)$. If this is correct then RSEL $\leq_\pi$ RQDE cannot hold.

(2) Because of lemma 2 RTTW is complete for NSPACE(log n) and there exists a set $L_1$ which is acceptable by some nondeterministic one-way reversal-bounded counter automaton such that $L_1 \equiv L_{RSEL}(\log n)$. We conjecture that no language acceptable by a reversal bounded counter automaton can be complete for NSPACE(log n). (Note that such an automaton cannot get any information from its storage tape during its computation. It changes its storage tape only by means of its finite memory and only at the end of its computation it asks whether it was able to generate the number 0 in this way). If $L_1$ is not complete for NSPACE(log n) then RTTW $\leq_\pi$ RSEL cannot hold.

(3) $L_{RTTW}(f)$ is accepted by an automaton which gets from its storage tape only the information whether it is empty or not empty. In the case $f(n) = \log n$ such automata accept languages which are complete for NSPACE(log n). This is true since the number of different storage inscriptions is bounded polynomially in n and we can store the whole flow of information in one string which is bounded polynomially in n (this is just the proof that the graph accessibility problem is complete for NSPACE(log n), [12]). But if $\lim_{n\to\infty} f(n)/\log n = \infty$ then the number of possible storage inscription grows faster than any polynomial in n and therefore this method is not applicable. We believe that for $\lim_{n\to\infty} f(n)/\log n = \infty$ the language $L_{RTTW}(f)$ is not complete for NPTIMESPACE(f) and this implies that RLSE is not reducible to RTTW.

Finally we want to mention that there is annother natural class (let us call this class SPP) such that $RPP \subset SPP \subset PP$ and $R \in SPP \Rightarrow L_R = \{u \not{c} v \mid (u,v) \in R\} \in NP$. R belongs to SPP iff R is accepted by a nondeterministic auxiliary pushdown automaton (see [4]) within polynomial time and the simultaneous space bound $\max\{\log n, m\}$. All the problems we looked at belonged to RPP. It would be interesting to find a natural problem which seems not to belong to RPP but which belongs to SPP. It would be even more interesting to find a natural problem which is complete for SPP.

Acknowledgement: We want to thank Hal Sudborough and Oliver Vornberger for many helpful discussions and for the careful reading of this manuscript.

References:

1. Ausiello, G., A. Marchetti-Spaccamela and M. Protasi, Toward a unified approach for the classification of NP-complete optimization problems, Proc. Frege-Conference 1979, Jena, GDR
2. Baker, B.S. and R.V. Book, Reversal-bounded multipushdown machines, J. Comp. Syst. Sci. 8(1974), 315-332
3. Conway, R.W., W.L. Maxwell and L.W. Miller, Theory of Scheduling, Addison-Wesley, Reading, Mass., 1967



4. Cook, S.A. Characterizations of Pushdown Machines in Terms of Time-Bounded Computers, J. Ass. Comp. Mach. 18 (1971), 4 - 18
5. Garey, M.R. and D.S. Johnson, "Strong" NP-Completeness Results: Motivation, Examples and Implications, J.Ass. Comp. Mach. 25 (1978), 499 - 5o8
6. Garey, M.R. and D.S. Johnson, Computers and Intractability, A Guide to the Theory of NP-completeness, W.H. Freeman and Company, San Francisco, 1979
7. Monien, B,. Connections between the LBA problem and the knapsack problem, Proc. Frege-Conference 1979, Jena, GDR
8. Monien, B., Scheduling problems and space bounded computations, in preparation
9. Monien, B. and I.H. Sudborough, Bounding the bandwidth of NP-complete problems, in preparation
1o. Paz, A. and S. Moran, Non-deterministic polynomial optimization problems and their approximation, Lecture Notes Comp. Sci. 52, 37o - 379, Springer Verlag Berlin-Heidelberg-New York, 1977
11. Sahni, S.K. Algorithms for Scheduling Independent Tasks, J.Ass. Comp. Mach. 23 (1976), 116 - 127
12 Savitch, W.J., Relationships between nondeterministic and deterministic tape complexities, J. Comp. Syst. Sci. 4(197o), 177 - 192