DIMACS Technical Report 97-78
December 1997

Remarks on the equivalence problem for PL-sets

by

Bhaskar DasGupta[1,2]
Department of Computer Science
Rutgers University
Camden, NJ 08102
Email: bhaskar@crab.rutgers.edu

[1]Visitor, DIMACS Center
[2]This result is part of an ongoing research with Prof. Eduardo Sontag on investigating PL-sets and I would like to thank him for his help

DIMACS is a partnership of Rutgers University, Princeton University, AT&T Labs-Research, Bell Labs and Bellcore.

DIMACS is an NSF Science and Technology Center, funded under contract STC–91–19999; and also receives support from the New Jersey Commission on Science and Technology.

ABSTRACT

The following problem arises in connection with checking the equivalence piecewise-linear (PL) sets. Consider two-variable polynomials with non-negative integer coefficients, and given two such polynomials, decide if they are equivalent to each other modulo the following equalities $x = 2x + 1$, $y^2 = 2y^2 + y$ and $y = x + y + 1$. In this paper, we show that this problem can be solved in polynomials time in both the unit-cost and the logarithmic-cost model.

1 Introduction and Basic Definitions

Let $\mathcal{S}$ denote the collection of all polynomials of two variables with *non-negative* integer coefficients. Let $\sigma = \{\sigma_1, \sigma_2, \sigma_3\}$ denote the following set of three equalities:

$$\begin{aligned} \sigma_1 &: \quad x = 2x + 1 \\ \sigma_2 &: \quad y^2 = 2y^2 + y \\ \sigma_3 &: \quad x = x + y + 1 \end{aligned}$$

Given two polynomials $P(x,y), Q(x,y) \in \mathcal{S}$, each of same degree n, we are interested in developing efficient algorithm for the following problem:

> Is $P(x,y)$ equivalent to $Q(x,y)$ modulo the equalities in σ? That is, is there a sequence of polynomials $R_0(x,y) = P(x,y), R_1(x,y), R_2(x,y), \ldots, R_t(x,y) = Q(x,y)$, such that every $R_i(x,y) \in \mathcal{S}$ and for every $i > 0$, there is a decomposition $R_i(x,y) = R_i'(x,y) + R_i''(x,y)$ and $R_{i-1}(x,y) = Q(x,y) + R_i''(x,y)$, with $Q(x,y), R_i'(x,y), R_i''(x,y) \in \mathcal{S}$, such that $R_i'(x,y)$ is obtained from $Q(x,y)$ by applying one of the equalities in σ (in either direction)?

This problem arises in connection with checking the equivalence of piecewise-linear (PL) sets; for example, see [1].

Let us use the notation $P(x,y) =_\sigma Q(x,y)$ to denote two polynomials equivalent in the above sense. Let $\neq_\sigma$ denote the negation of the relation $=_\sigma$.

Example: $2x^2 + 2y^2 + y =_\sigma 5x^2 + 4x + y + 1$, since

$$2x^2 + 2y^2 + y = x^2 + x^2 + 2y^2 + y =_\sigma x^2 + (2x+1)^2 + 2y^2 + y =_\sigma 5x^2 + 4x + y + 1$$

whereas $y^2 + y \neq_\sigma x^2$.

Notice that $P(x,y) =_\sigma Q(x,y)$ if and only if $Q(x,y) =_\sigma P(x,y)$, since each equality in σ can be applied in either direction, and hence $=_\sigma$ is an equivalence relation on $\mathcal{S}$. A consequence of $=_\sigma$ being an equivalence relation is:

Observation 1.1 *Let $P =_\sigma P'$. Then, $P =_\sigma Q$ if and only if $P' =_\sigma Q$.*

It was shown in [1] that the decision procedure is decidable. However, no efficient algorithm was known for this procedure (or, a proof that the problem is intrinsically hard, i.e. NP-hard). The purpose of this paper is to give a polynomial-time algorithm to decide if $P(x,y) =_\sigma Q(x,y)$. Regarding the precise meaning of polynomial-time computation, there are at least two models of complexity possible. The first, the *unit cost model* of computation, is intended to capture the algebraic complexity of the problem; in that model, each arithmetic and comparison operation on two real numbers is assumed to take unit time. An alternative, the *logarithmic cost model*, is closer to the notion of computation in the usual Turing machine sense; in this case one assumes that each coefficient of the two gives polynomials is an integer with at most B bits, each arithmetic and comparison operation on two B bit integers takes $O(B)$ time, and the time involved in the decision procedure is required to be polynomial on B as well. For convenience, we will just write $O(\alpha)$ time to denote a running time of $O(\alpha)$ in the unit-cost model or a running time of $O(\alpha B)$ in the logarithmic-cost model.

The main theorem of this paper is the following.

Main Theorem. Given any two polynomials $P, Q \in \mathcal{S}$, whether $P =_\sigma Q$ or not can be decided in $O(n^2)$ time. Moreover, if $P =_\sigma Q$, then a sequence of valid operations transforming P to Q can be computed in $O(n^2)$ time.

The proof of the main theorem extends over the following few sections. Notice that the time taken by our algorithm for the unit-cost model is optimal within a constant factor in the worst case, since the polynomials have $O(n^2)$ coefficients. Similarly, the time for the logarithmic-cost model is also optimal within a constant factor in the worst case.

We assume that both the given polynomials have the same degree, since application of any equality in σ preserves the degree of the polynomial. Unless otherwise stated explicitly, the following notations and conventions will be used throughout the rest of the paper. $P(x,y) = \sum_{i,j\geq 0\,;\,i+j\leq n} p_{i,j}x^iy^j$ and $Q(x,y) = \sum_{i,j\geq 0\,;\,i+j\leq n} q_{i,j}x^iy^j$ denote the two degree n polynomials involved in the decision procedure (with $p_{i,j}, q_{i,j} \geq 0$ for all i and j, and there exist some $i', i'', j', j'', i'+j' = j'+j'' = n$ such that $p_{i',j'}, q_{i'',j''} > 0$). Sometimes we will omit the variables for clarity and simply refer to the two polynomials as P and Q. For the logarithmic cost model, we will also assume that each $p_{i,j}$ and $q_{i,j}$ is at most B bits long for some $B > 0$. For two polynomials P and Q, define $P - Q = \sum_{i,j\geq 0\,;\,i+j\leq n} (p_{i,j-q_{i,j}})x^iy^j$. Notice that the coefficients of $P - Q$ can be negative. Let α_i and β_i denote the left-side and right-side, respectively, of the equality $\sigma_i \in \sigma$ (i.e., α_1 is x, β_1 is $2x+1$, etc.). Finally, the notation *polylog*(x) denotes the function $\log^c x$ for some integer $c > 0$.

2 Some Preliminary Results

It is not too difficult to see that if $P =_\sigma Q$, then there is a sequence of transformation of P to Q such that every polynomial in the sequence (except the first one, i.e. P) can be obtained from the previous one by replacing a single occurrence of the left-side (resp. right-side) of some equality $\sigma_i \in \sigma$, possibly multiplied by other terms, by the right-side (resp. by the left-side) of σ_i. More precisely, one can show the following proposition, whose proof is implicit in [1].

Proposition 2.1 *Assume $P =_\sigma Q$. Then, there is a sequence of transformation of P to Q such that every polynomial in the sequence (except the first one, i.e. P) can be obtained from the previous one by replacing $\gamma\alpha_i$ (resp. $\gamma\beta_i$) by $\gamma\beta_i$ (resp. $\gamma\alpha_i$) for some $1 \leq i \leq 3$, where $\gamma \in \mathcal{S}$.*

Let $R = \sum_{i,j\geq 0\,;\,i+j\leq n} r_{i,j}x^iy^j$ be an intermediate polynomial in a transformation from P to Q (initially, $R = P$). Then, *one* application of an equality from σ changes the coefficients of

R in one of the following manner:

(a) Substituting x by $2x+1$ (forward direction of rule σ_1).

Assume that the substitution involves the term $r_{i,j}x^iy^j$ (with $i, r_{i,j} \neq 0$) and is of the form (for some $0 < c \leq r_{i,j}$):

$$r_{i,j}x^iy^j =_\sigma (r_{i,j}-c)x^iy^j + cx^{i-1}y^j(2x+1) = (r_{i,j}+c)x^iy^j + cx^{i-1}y^j$$

This is equivalent to increasing both $r_{i,j}$ and $r_{i-1,j}$ by c.

(b) Substituting $2x+1$ by x (backward direction of rule σ_1).

This is the reverse of (a) and hence is equivalent to decreasing $r_{i,j}$ and $r_{i-1,j}$ by c as long as $r_{i,j} > c$.

(c) Substituting y^2 by $2y^2+y$ (forward direction of rule σ_2).

By a similar reasoning to as in (a), this is equivalent to increasing both $r_{i,j}$ and $r_{i,j-1}$ by c, as long as $j > 1$, $r_{i,j} > 0$ **and** $0 < c \leq r_{i,j}$.

(d) Substituting $2y^2+y$ by y^2 (backward direction of rule σ_2).

This is the reverse of (c) and hence is equivalent to decreasing both $r_{i,j}$ and $r_{i,j-1}$ by c, as long as $j > 1$ **and** $r_{i,j} > c$.

(e) Substituting x by $x+y+1$ (forward direction of rule σ_3).

By a similar reasoning to as in (a), this is equivalent to increasing both $r_{i-1,j}$ and $r_{i-1,j+1}$ by c, as long as $i, r_{i,j} \neq 0$ **and** $j < n$.

(f) Substituting $x+y+1$ by x (backward direction of rule σ_3).

This is the reverse of (e) and hence is equivalent to decreasing both $r_{i-1,j}$ and $r_{i-1,j+1}$ by c, as long as $i, r_{i,j} \neq 0$, $j < n$ and $r_{i-1,j}, r_{i-1,j+1} \geq c$.

Now, notice that the restriction $c \leq r_{i,j}$ can be dropped in (a) and (c) above. Indeed, assume that we want to increase $r_{i,j}$ by $c > r_{i,j}$. Let $\eta r_{i,j} \leq c < (\eta+1)r_{i,j}$ for some positive integer $\eta \geq 1$. Then, we first increase $r_{i,j}$ to $2r_{i,j}$, then increase $2r_{i,j}$ to $3r_{i,j}$ and continue this way until we arrive at $\eta r_{i,j}$ and then, if $c - \eta r_{i,j} > 0$, we increase $\eta r_{i,j}$ by $c - \eta r_{i,j}$.

Definition 2.1 *A* **valid operation** *on the coefficients of a polynomial $R \in \mathcal{S}$ is one of the following operations:*

- *For some $r_{i,j} > 0$, do one of the following:*

  Move I(a): *if $i > 0$, increase both $r_{i,j}$ and $r_{i-1,j}$ by an arbitrary integer $c > 0$.*

  Move I(b): *if $j > 1$, increase both $r_{i,j}$ and $r_{i,j-1}$ by an arbitrary integer $c > 0$.*

Move I(c): *if $i > 0$ and $j < n$, increase both $r_{i-1,j}$ and $r_{i-1,j+1}$ by an arbitrary integer $c > 0$.*

Move I(d): *if $i > 0$, $j < n$, $r_{i-1,j}, r_{i-1,j+1} \geq c$ for some arbitrary integer $c > 0$, decrease both $r_{i-1,j}$ and $r_{i-1,j+1}$ by c.*

- *For some $a(i,j) > c$ ($c > 0$ is an arbitrary integer):*

Move II(a): *if $i > 0$, decrease both $r_{i,j}$ and $r_{i-1,j}$ by c.*

Move II(b): *if $j > 1$, decrease both $r_{i,j}$ and $r_{i,j-1}$ by c.*

Combining everything, the problem to decide if $P =_\sigma Q$ can be alternatively formulated as follows.

INSTANCE: Two polynomials $P, Q \in \mathcal{S}$.

QUESTION: Is there a sequence of zero or more valid operations that changes the coefficients of P such that at the end $p_{i,j} = q_{i,j}$ for all i and j?

Definition 2.2 *For a coefficient $r_{i,j}$ of a polynomial $R \in \mathcal{S}$, the* **neighbors** *of $r_{i,j}$, denoted by $N(r_{i,j})$, are the (at most 4) coefficients $r_{i',j'}$ such that $|i - i'| + |j - j'| = 1$.*

We will, from now on, interchangeably look at either the original problem or at the above formulation of the problem through valid operations, whichever is more convenient for our purpose.

3 Some results on Modifying Coefficients

In this section, we discuss some results and techniques to modify the coefficients $r_{i,j}$ of $R \in \mathcal{S}$ by valid operations, where R is assumed to be a polynomial of degree n obtained by performing zero or more valid operations on P.

Lemma 3.1 *Let $r_{i,j} > 0$ for some indices i and j and let $K > 0$ and $L < r_{i,j}$ be two arbitrary positive integers. Let $M = r_{i,j} - L > 0$. Then, there is a sequence of at most $O(n)$ valid operations changing the coefficients of R such that the final values of the coefficients, $r'_{i,j}$, satisfy one or more of the following:*

(a) *If $i > 0$, then $r'_{i',j} \geq K$ for all $i'' \leq i' \leq i$, where i'' is any arbitrary integer satisfying $0 \leq i'' \leq i$.*

(b) *If $j > 1$, then $r'_{i,j'} \geq K$ for all $j'' \leq j' \leq j$, where j'' is any arbitrary integer satisfying $1 \leq j'' \leq j$.*

(c) *If $i + j = n$, then $r'_{i',j'} \geq K$ for all $i' + j' = n$, $i \leq i' \leq i''$, $j \leq j' \leq j''$, where i'', j'' are arbitrary integers satisfying $i'' + j'' = n$, $i \leq i'' \leq n$, $j \leq j'' \leq n$.*

(d) *If $i > 0$, then $r'_{i',j} = r_{i',j}$ for all $i'' < i' < i$, $r'_{i,j} = L$ and $r'_{i'',j}$ is increased or decreased by M from its old value $r_{i'',j}$, where i'' is any arbitrary integer satisfying $0 \leq i'' \leq i$ and $r_{i'',j} \geq L$.*

(e) *If $i > 0$, then $r'_{i',j} = r_{i',j}$ for all $i'' < i' < i$, $r'_{i,j} = r_{i,j} + K$ and $r'_{i'',j}$ is increased or decreased by K from its old value $r_{i'',j}$, where i'' is any arbitrary integer satisfying $0 \leq i'' \leq i$ and $r_{i'',j} \geq K$.*

Proof: **(a)** Since $r_{i,j} > 0$, we can use the following sequence of valid operations, each of type I(a): increase $r_{i,j}$ and $r_{i-1,j}$ by K; then, increase $r_{i-1,j}$ and $r_{i-2,j}$ by K; and continue this way until we have increased the pair $r_{i''+1,j}$ and $r_{i'',j}$ by K (if $i'' = i$, we just increase the pair $r_{i-1,j}$ and $r_{i-2,j}$ by K). Obviously, we need at most n valid operations.
(b) Similar to (a) above (we need to use valid operations of type I(b) this time).
(c) There is nothing to do if $i = 0$ and $j = n$. Otherwise, since $r_{i,j} > 0$, we can use the following sequence of valid operations, each of type I(c): first, increase the pair $r_{i-1,j}$ and $r_{i-1,j+1}$ by K; next, since $r_{i-1,j+1} > 0$, increase the pair $r_{i-2,j+1}$ and $r_{i-2,j+2}$ by K; and continue this way until we have increased the pair $r_{i'',j''-1}$ and $r_{i'',j''}$ by K (if $i'' = i$, then we just increase the pair $r_{i-1,j}$ and $r_{i-1,j+1}$ by K). Again, we perform at most n valid operations.
(d) Since $r_{i,j} > 0$, we can use the following sequence of valid operations:

(i) Use a sequence of at most n valid operations as described in part (a) (with $K = M$) to ensure that $r_{i',j} > M$ for all $i'' < i' \leq i$. Notice that this step increases the value of $r_{i,j}$ by M.

(ii) Use the following sequence of alternately decreasing (type II(a)) and increasing (type I(a)) valid operations: decrease $r_{i,j}$ and $r_{i-1,j}$ by M; next, increase $r_{i-1,j}$ and $r_{i-2,j}$ by M; next, decrease $r_{i-2,j}$ and $r_{i-3,j}$ by M; and so forth, until we have performed valid operation on the pair $r_{i''-1,j}$ and $r_{i'',j}$.

(iii) Finally, use the reverse sequence of valid operations as used in (i) to decrease the coefficients $r_{i',j} > M$ for $i'' < i' \leq i$ (notice that this step decreases the value of $r_{i,j}$ by M).

(e) Similar to (d) above. ■

4 Few Necessary Conditions for Transformability

In this section, we discuss some necessary (and, under certain conditions, sufficient) conditions for transforming polynomial P to Q. First, we state an easy necessary condition.

Proposition 4.1 *$P =_\sigma Q$ implies $\sum_i \sum_j p_{i,j} = \sum_i \sum_j q_{i,j} \pmod 2$.*

Proof: Every valid operation on a polynomial increases or decreases the total sum of all the coefficients of the polynomial by an even integer. ■

Definition 4.1 *Given any two polynomials $P', Q' \in \mathcal{S}$, let $R' = P' - Q' = \sum_{i,j\geq 0\,;\,i+j\leq n} r'_{i,j}x^iy^j$. For $i,j \geq 0$, $i+j=n$, let $D_{i,j} = \sum_{s=0}^{j}(-1)^s r'_{i,j}$. Define the* **characteristic function** *$\Lambda : R' \mapsto \mathcal{N}$ to be the following function:*

$$\Lambda(R) = D_{0,n} + D_{1,n-1} + D_{2,n-2} + \cdots + D_{n,0}$$

Figure 1 illustrate how $\Lambda(R)$ is calculated when $n = 5$. Notice that the sign of every coefficient of R' is *exactly the opposite* as that of any of its neighbor. $\Lambda(R)$ can be computed trivially in $O(n^2)$ time.

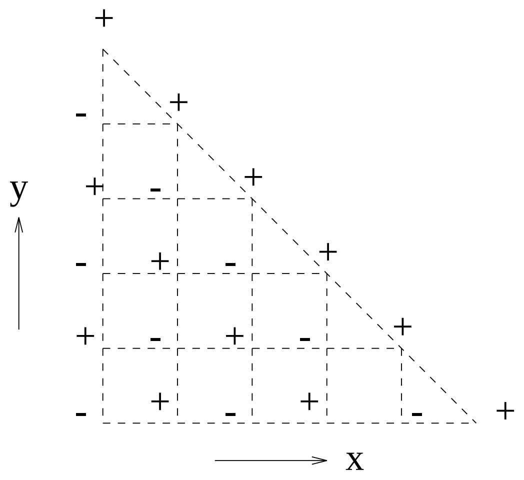


Figure 1: Illustration of how $\Lambda(R)$ is calculated for a degree 5 polynomial $R' = P' - Q'$. The grid-point with coordinates (i,j) stores the coefficient $r_{i,j}$. A + sign (resp. – sign) in the figure for the grid-point for $r_{i,j}$ indicates that this element is added (resp. subtracted) in the summation of $\Lambda(R)$.

Lemma 4.1 *Let $R = P - Q$. Then, $\Lambda(R) \neq 0$ implies $P \neq_\sigma Q$.*

Proof: We prove by showing that if P is changed to P' by a single valid operation, then $\Lambda(P-Q) = \Lambda(P'-Q)$. Since $\Lambda(Q-Q) = 0$, this will prove (by induction on the length of the transformation sequence) that the value of $\Lambda(P-Q)$ must be zero.

Let $R = \sum_{i,j\geq 0\,;\,i+j\leq n} r_{i,j}x^iy^j$. Consider a valid operation changing P. An inspection of Definition 2.1 shows that a valid operation either changes both $r_{i,j}$ and $r_{i-1,j}$ by the same amount or it changes both $r_{i,j}$ and $r_{i,j-1}$ by the same amount, for some appropriate indices i and j. In the former case, since $r_{i,j}$ and $r_{i-1,j}$ have opposite signs in the expression for $\Lambda(R)$, the value of $\Lambda(R)$ remains unchanged. In the later case also, since $r_{i,j}$ and $r_{i,j-1}$ have opposite signs in the expression for $\Lambda(R)$, the value of $\Lambda(R)$ remains unchanged. ■

Corollary 4.1 *Let P and Q differ in exactly one coefficient. Then, $P \neq_\sigma Q$.*

Unfortunately, the conditions stated in Proposition 4.1 or in Lemma 4.1 are not sufficient. For example, if $P = y + 1$ and $Q = 2y + 2$, then $\Lambda(P - Q) = 0$ and $\sum_i \sum_j p_{i,j} = \sum_i \sum_j q_{i,j}$ (mod 2), but $P \neq_\sigma Q$. However, in the one-variable case (Section 5) or in the two-variable case provided $p_{0,n}, q_{0,n} > 0$ (Section 6.1), the condition stated in Lemma 4.1 is sufficient also. That is, in particular, one could prove the following result.

Lemma 4.2 *If $p_{0,n}, q_{0,n} > 0$, then $\Lambda(P - Q) = 0$ implies $P =_\sigma Q$.*

The proof of Lemma 4.2 follows easily by a careful inspection of the proofs of Theorem 6.1 and the main theorem.

Lemma 4.3 *Consider the diagonal coefficients of P and Q, i.e., the elements $p_{n-j,j}, q_{n-j,j}$ for $0 \leq j \leq n$. Let j' be the least integer such that $p_{n-j',j'} > 0$. Then, if $P =_\sigma Q$, then j' is also the least integer such that $q_{n-j',j'} > 0$.*

Proof: Since $p_{n-j,j} = 0$ for all $j < j'$, there is no valid operation that can decrease the value of $p_{n-j',j'}$ to zero, hence $q_{n-j',j'} > 0$. Moreover, there is no valid operation that can increase the value of any $p_{n-j,j}$ (for $j < j'$) to a non-zero value. Hence j' is the least integer such that $q_{n-j',j'} > 0$. ■

Lemma 4.4 *Let P and Q be two polynomials such that P and Q differ in exactly three coefficients in the following manner: all the coefficients of P and Q are identical except possibly at most three pairs of coefficients $(p_{i,j}, q_{i,j})$, $(p_{i-1,j}, q_{i-1,j})$ and $(p_{i,j-1}, q_{i,j-1})$ for some indices i, $j > 0$, $i + j = n$. Furthermore, assume that $p_{i,j}, q_{i,j} > 0$. Then, we can decide in $O(1)$ time if $P =_\sigma Q$. Moreover, if $j > 1$, then $\Lambda(P - Q) = 0$ implies $P =_\sigma Q$ (i.e., the condition in Lemma 4.1 is sufficient also).*

Proof: If exactly one of the three pairs of coefficients differ, then $\Lambda(P - Q) \neq 0$ and by Corollary 4.1, $P \neq_\sigma Q$. Next, assume exactly two of those three coefficients differ. We may have the following 3 cases:

Case 1. $(p_{i,j} \neq q_{i,j})$ **and** $(p_{i-1,j} \neq q_{i-1,j})$. If $P =_\sigma Q$, then, by Lemma 4.1, $\Lambda(P - Q) = 0$, that is, $p_{i,j} - q_{i,j} = p_{i-1,j} - q_{i-1,j} = c$ for some integer $c \neq 0$. If $c < 0$, increase both $p_{i,j}$ and $p_{i-1,j}$ by c to transform P to Q. If $c > 0$, then since $q_{i,j} > 0$, decrease both $p_{i,j}$ and $p_{i-1,j}$ by c to transform P to Q.

Case 2. $(p_{i,j} \neq q_{i,j})$ **and** $(p_{i,j-1} \neq q_{i,j-1})$. This is similar to Case 1 if $j > 1$. Now, consider the case when $j = 1$. If $p_{n,0} > 0$ then again it is similar to Case 1. So, consider the case when $p_{n,0} = 0$. If $p_{n-1,0} = 0$, then $P \neq_\sigma Q$, since there is no valid operation that can increase $p_{n-1,0}$ to a non-zero value. Similarly, if $q_{n-1,0} = 0$, then $P \neq_\sigma Q$. Hence, we need to consider the case when $p_{n-1,0}, q_{n-1,0} > 0$. If there is no index $0 < i < n - 1$ such that $p_{i,0} > 0$, then again there is no sequence of valid operations that can change the value of $p_{n-1,0}$ and hence $P \neq_\sigma Q$. Hence, assume that there is such an index i. Now, first, increase both $p_{i-1,0}$ and $p_{i-1,1}$ such that $p_{i-1,0}, p_{i-1,1} > \max\{|p_{n,0} - q_{n,0}|, |p_{n-1,0}, q_{n-1,0}|\}$. Now, use Lemma 3.1 (parts (d) and (e)) to transform P such that P differs from Q in the two coefficients $p_{i-1,0}$ and $p_{i-1,1}$. Since $p_{i,0} > 0$, this is now similar to Case 1.

Case 3. $(p_{i,j-1} \neq q_{i,j-1})$ **and** $(p_{i-1,j} \neq q_{i-1,j})$. If $P =_\sigma Q$, then $\Lambda(P - Q) = 0$, hence $p_{i,j-1} - q_{i,j-1} = -(p_{i-1,j} - q_{i-1,j}) = c$ for some integer $c \neq 0$.

First, consider the case when $j > 1$. If $c > 0$, first increase both $p_{i,j}$ and $p_{i-1,j}$ by c, then decrease both $p_{i,j}$ and $p_{i,j-1}$ by c. If $c < 0$, first increase both $p_{i,j}$ and $p_{i,j-1}$ by c, then decrease both $p_{i,j}$ and $p_{i-1,j}$ by c. In either case P is transformed to Q.

Now, consider the case $j = 1$. If $p_{n,0} > 0$, this is similar to the case for $j > 1$. So, assume that $p_{n,0} = 0$. If $c > 0$, first increase both $p_{n-1,1}$ and $p_{n-2,1}$ by c and now this is Case 2. If $c < 0$, then, if $p_{n-1,0} = 0$ then $P \neq_\sigma Q$ (since $q_{n-1,0} > 0$ and there are no valid operations to increase the value of $p_{n-1,0}$). So, assume that $p_{n-1,0} > 0$. Now, first, increase both $p_{n-1,0}$ and $p_{n-2,0}$ by c, then decrease both $p_{n-2,1}$ and $p_{n-2,0}$ by c. This transforms P to Q.

Next, consider the case in which all the three pairs of coefficients differ. Let $c = p_{i,j} - q_{i,j}$, $c_1 = p_{i,j-1} - q_{i,j-1}$ and $c_2 = p_{i-1,j} - q_{i-1,j}$, $c_1, c_2, c_3 \neq 0$. If $P =_\sigma Q$, then $\Lambda(P - Q) = 0$, hence $c = c_1 + c_2$. Now, there are two sub-cases:

Case I. $c > 0$. If $c_2 > 0$, then decrease both $p_{i-1,j}$ and $q_{i-1,j}$ by $\min\{c, c_2\}$. If $c_2 < 0$, then increase both $p_{i-1,j}$ and $q_{i-1,j}$ by c_2.

Case II. $c < 0$. Increase both $p_{i-1,j}$ and $q_{i-1,j}$ by c.

After either Case I or Case II, we have transformed P such that at most two of its coefficients differ from Q, and the solution for this has been already given.

An inspection of this proof also shows that if $j > 1$, then we can always transform P to Q provided $\Lambda(P - Q) = 0$. ■

Before stating the next lemma, we need a few well-known definitions from computational geometry.

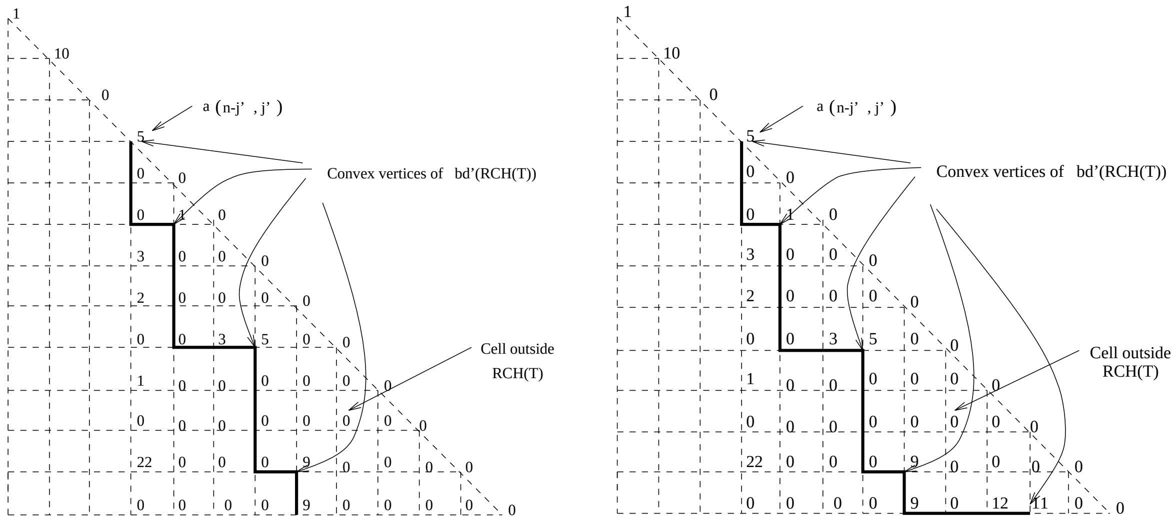
Definition 4.2 *A rectilinear polygon in the plane is a connected polygon (without holes) whose sides are parallel to the two coordinate axes. A rectilinear polygon is rectilinearly-convex (RC, for short) if, for any two points with coordinates (x_0, y_0) and (x_1, y_1) inside the polygon with either $x_0 = x_1$ or $y_0 = y_1$, the line segment connecting them is also completely inside the polygon (a point on the boundary is considered to be inside the polygon). Finally, given any set of points T on the plane, the rectilinear-convex-hull of T (denoted by $RCH(T)$) is the smallest RC polygon containing T. The boundary of $RCH(T)$ will be denoted by $bd(RCH(T))$. A vertex on $RCH(T)$ is a convex vertex if it is produced by the intersection of two consecutive edges of $RCH(T)$ which forms a $90°$ angle inside the interior of $RCH(T)$.*

Given a set T of points on a grid of size $n \times n$, it is easy to compute $RCH(T)$ in $O(n^2)$ time Let us consider a grid of size $n \times n$ in which the grid-point with coordinate (i, j) (for $0 \leq i, j \leq n-1$) contains the value $p_{i,j}$. We will refer to this grid-point by $a(i, j)$. Let $T = \{a(i, j) \mid a(i, j) \text{has a non-zero value}\}$.

Definition 4.3 *Let j' be the least integer such that $p_{n-j',j'} > 0$. Let $bd'(RCH(T))$ be the portion of $bd(RCH(T))$ that we traverse as we move from $a(n-j', j')$ until we reach the horizontal axis. If there is at least one grid-point $a(0, j)$ to the right of $bd'(RCH(T))$, then extend $bd'(RCH(T))$ such that $bd'(RCH(T))$ includes the rightmost non-zero grid-point on the horizontal axis (see Fig. 2 for illustrations).*

Next, we define the convex vertices of $bd'(RCH(T))$ in the following manner. Extend $bd'(RCH(T))$ to an infinite polygon by drawing a horizontal line through $a(n-j', j')$ to the left and by drawing a vertical line downwards from the point on $bd'(RCH(T))$ on the horizontal axis. Let a convex vertex of $bd'(RCH(T))$ refer to a convex vertex of this infinite polygon that is also a vertex of $bd'(RCH(T))$ (see Fig. 2 for illustrations).

Finally, a grid-point $a(r, s)$ is considered to be **outside** *of $bd'(RCH(T))$ provided $n-j' \leq r \leq n$, $0 \leq s \leq j'$, $0 \leq r+s \leq n$, and $a(r, s)$ is outside $RCH(T)$ (a point on $bd'(RCH(T))$ will be treated as inside $RCH(T)$).*



(a) (b)

Figure 2: Illustration of $bd'(RCH(T))$ and convex vertices for $bd'(RCH(T))$ for two different polynomials. The value contained by a grid-point is shown beside it. Values not shown are assumed to be zero. Note that in (a) the grid-point on the horizontal axis is **not** counted as a convex vertex on $bd'(RCH(T))$. Also, note that the grid-point $a(n-j', j')$ is always counted as a convex vertex of $bd'(RCH(T))$.

Lemma 4.5 *If $P =_\sigma Q$, then the following two statements hold:*

(a) *If $a(r,s)$ is a convex vertex of $bd'(RCH(T))$, then $q_{r,s} > 0$.*

(b) *Let $a(r,s)$ be any grid-point outside of $bd'(RCH(T))$. Then, $q_{r,s} = 0$.*

Proof: Let j' be as defined in Definition 4.3. Note that if $j' = 0$ then $bd'(RCH(T))$ consists of a single vertex $a(n,0)$ (which is also a convex vertex of $bd'(RCH(T))$). Part (a) then follows from Lemma 4.3, and part(b) holds trivially, since there is no grid-point outside of $bd'(RCH(T))$. So, assume that $j' > 0$.
(a) Since all the non-zero entries are either to the left or below $a(r,s)$, there is no valid operation that can decrease $a(r,s)$ to zero.
(b) Since all the non-zero entries are either to the left or below $a(r,s)$, there is no valid operation that can increase $a(r,s)$ to a non-zero value. ∎

5 The One-Variable Case

In this section we consider the simpler case when both P and Q are polynomials in only one variable x (i.e., $p_{i,j} = q_{i,j} = 0$ for all $j > 0$). This simpler case is important because a similar technique for its solution will be applied repeatedly (with appropriate modifications) to solve the original problem. For notational convenience, let us denote $p_{0,j}$ (resp. $q_{0,j}$) simply by p_j (resp. q_j). Notice that $p_n, q_n > 0$.

Theorem 5.1 *In the one-variable case, $P =_\sigma Q$ if and only if $\Lambda(P-Q) = 0$. Moreover, if $P =_\sigma Q$, then a sequence of valid operations transforming P to Q can be computed in $O(n)$ time.*

Proof: It turns out that if $P =_\sigma Q$, then P can be transformed to Q by using only the equality σ_1. The proof will be constructive and will produce an $O(n)$ time algorithm to give a sequence of transformations from P to Q (or, report that $P \neq_\sigma Q$). The algorithm works in two steps:

Step 1: Transform P to another polynomial P' using the equality σ_1 such that P' differs from P in *at most* one coefficient.

Step 2: Use Corollary 4.1 and Observation 1.1 to conclude that $P =_\sigma Q$ if and only if P' is the same as Q.

Since Step 2 is trivial, we concentrate on Step 1. We assume $\Lambda(P-Q) = 0$ (since otherwise $P \neq_\sigma Q$). We specify how to modify $p_0, p_1, p_2, \ldots, p_{n-2}$, in that order, using valid operations corresponding to the equality σ_1, such that at the end we have $p_0 = q_0$, $p_1 = q_1$,…,$p_{n-2} = q_{n-2}$. First, since $p_n > 0$, use Lemma 3.1 (part (a)) to increase $p_n, p_{n-1}, \ldots, p_0$ by 1. This takes $O(n)$ time. Notice that, after this step, $p_i > 0$ for all indices i.

Inductively, assume that before the i^{th} step of coefficient-correction ($i = 0, 1, 2, \ldots, n-2$), we have already corrected the coefficients $p_0, p_1, \ldots, p_{i-1}$, possibly by modifying some of the coefficients $p_i, \ldots, p_n$, but **maintaining the following invariant**: $p_i, p_{i+1}, \ldots, p_n > 0$. We now show how to correct the current value of p_i (if $p_i \neq q_i$) and maintain the invariant that $p_{i+1}, p_{i+2}, \ldots, p_n > 0$. There are three cases to consider:

Case 1. $p_i < q_i$. Then, since $p_{i+1} > 0$, increase both p_i and p_{i+1} by $q_i - p_i$.

Case 2. $p_i > q_i$ and $p_{i+1} > (p_i - q_i)$. Then, decrease both p_i and p_{i+1} by $p_i - q_i$.

Case 3. $p_i > q_i$ and $p_{i+1} \leq (p_i - q_i)$. Then, (since $p_{i+2} > 0$) first increase both p_{i+1} and p_{i+2} by $(p_i - q_i) - p_{i+1} + 1$. Then, decrease both p_i and p_{i+1} by $p_i - q_i$.

Hence, we have changed the coefficients so that all but the leading two coefficients differ in the two polynomials. Finally, since $\Lambda(P - Q) = 0$, and its value does not change by any sequence of valid operations, we must have $p_n - q_n = p_{n-1} - q_{n-1}$. Also, remember that $q_n > 0$ and $q_{n-1} \geq 0$. If $p_n < q_n$, we increase both p_{n-1} and p_n by $p_n - q_n$, otherwise, if $p_n > q_n$, we decrease p_{n-1} and p_n by $p_n - q_n$. Hence, our complete algorithm is as follows:

```
if ∑_{i=0}^{n-1} (p_i − q_i) ≠ 0 then

   for i = n, n − 1, n − 2, ..., 1 do
      increase p_i and p_{i−1} by 1
   endfor

   for i = 0, 1, 2, ..., n − 2 do
      if p_i < q_i, increase p_i and p_{i+1} by q_i − p_i
      if p_i < q_i and p_{i+1} > (p_i − q_i), decrease p_i and p_{i+1} by p_i − q_i
      if p_i < q_i and p_{i+1} > (p_i − q_i) then
         increase p_{i+1} and p_{i+2} by (p_i − q_i) − p_{i+1} + 1
         decrease p_i and p_{i+1} by p_i − q_i
   endfor

   if p_n < q_n, increase p_{n−1} and p_n by p_n − q_n
   if p_n < q_n, decrease p_{n−1} and p_n by p_n − q_n

   if P is identical to Q then P =_σ Q; otherwise P ≠_σ Q.

else P ≠_σ Q.
```

Algorithm for the one-variable case.

It is clear that the total time taken by the above algorithm is $O(n)$. This completes the proof. ∎

6 The Two-variable Case (Original Problem)

Now, we consider the original problem when one or both of P and Q have non-zero coefficients of terms involving positive powers of y.

6.1 A Slightly Simpler Case: $p_{0,n} > 0$

Theorem 6.1 *Assume that $p_{0,n} > 0$. Then, $P =_\sigma Q$ if and only if $q_{0,n} > 0$* **and** *$\Lambda(P-Q) = 0$. Moreover, if $P =_\sigma Q$, then a sequence of valid operations transforming P to Q can be computed in $O(n^2)$ time.*

Proof: If $p_{0,n} > 0$ and $P =_\sigma Q$, then, by Lemma 4.3, $q_{0,n} > 0$. By Lemma 3.1 (part (c) followed by part (a)), if necessary, we can, by a sequence of at most $O(n)$ valid operations (taking $O(n)$ total time), change the coefficients of P such that $p_{i,j} > 0$ for all i and j.

Our algorithm has n steps. Inductively assume that before the j^{th} step of the algorithm ($j = 0, 1, \ldots, n$), we have ensured that $p_{i,j'} = q_{i,j'}$ for all $n-j < j' \leq n$, $0 \leq i \leq n$, $i+j' \leq n$, and **have maintained the following invariant**: $p_{i'',j''} > 0$, $0 \leq j'' \leq n-j$, $0 \leq i'' \leq n-j''$. Notice that the invariant trivially holds when $j = 0$.

Now, we describe the j^{th} step of our algorithm. Consider the coefficients $p_{0,n-j}, p_{1,n-j}, \ldots, p_{j,n-j}$. Apply the algorithm in the one-variable case to modify the coefficients such that $p_{i,n-j} = q_{i,n-j}$ for all $0 \leq i \leq j-2$. Finally, if $j \neq n$, we do the following:

Case 1. $j = 0$. Let $c = p_{0,n} = q_{0,n}$. If $c < 0$, since $p_{1,n-1} > 0$, increase both $p_{0,n}$ and $p_{0,n-1}$ by c. If $c > 0$ and $p_{0,n-1} \geq c$, then (since $p_{1,n-1} > 0$), decrease both $p_{0,n}$ and $p_{0,n-1}$ by c. If $c > 0$ and $p_{0,n-1} < c$, first increase both $p_{1,n-1}$ and $p_{0,n-1}$ by $c - p_{0,n-1}$, then decrease both $p_{0,n}$ and $p_{0,n-1}$ by c.

Case 2. $j > 0$. Let $c_1 = p_{j-1,n-j} - q_{j-1,n-j}$ and $c_2 = p_{j,n-j} - q_{j,n-j}$. Remember that $p_{j-1,n-j}, p_{j,n-j}, p_{j-1,n-j-1}, p_{j,n-j-1} > 0$. By using Lemma 3.1, if necessary, we can make sure that $p_{j-1,n-j-1}, p_{j,n-j-1} > \max\{c_1, c_2, 1\}$. We are done if $c_1 = c_2 = 0$. Otherwise, if $c_1 > 0$ (resp. $c_1 < 0$), decrease (resp. increase) both $p_{j-1,n-j}$ and $p_{j-1,n-j}$ by c_1. Then, if $c_2 > 0$ (resp. $c_2 < 0$), decrease (resp. increase) both $p_{j,n-j}$ and $p_{j,n-j}$ by c_2.

Hence, at the end of the n^{th} stage, we have modified P such that P differs from Q in at most two coefficients $p_{0,n-1}$ and $p_{0,n}$. Now, use the same argument as in the last part of the proof of Theorem 5.1 to conclude that if $\Lambda(P-Q) = 0$ then $P =_\sigma Q$.

It is clear that we use at most $O(n^2)$ arithmetic operations. Moreover, every number during the computation has at most $O(B)$ bits. So, we take $O(n^2)$ time. ■

6.2 The Case When $p_{0,n} = 0$

Now, we prove the general case when $p_{0,n} = 0$.

Proof of Main Theorem: Assume that $p_{0,n} = 0$, since otherwise the proof has already been given in Theorem 6.1. Let $j' > 0$ be the least integer such that $p_{n-j',j'} > 0$. By Lemma 4.3, j' is also the least integer such that $q_{n-j',j'} > 0$. Consider the set T and the same notations as used in Lemma 4.5. Let $a(r_1, n-r_1), a(r_2, n-r_2), \ldots, a(r_\ell, n-r_\ell)$ be the $\ell > 0$ convex vertices of $bd'(RCH(T))$, where $r_1 = n - j'$, $r_1 < r_2 < \cdots < r_\ell$.

When $\ell = 1$:

First, consider the simpler case when $\ell = 1$. Then, by Lemma 4.5, $p_{i,j} = q_{i,j} = 0$ for all $i > n - j'$, $j < j'$, $0 \le i + j \le n$. Notice that either $p_{0,j'} = q_{0,j'} = 0$ or $p_{0,j'}, q_{0,j'} > 0$, since there are no valid operations that can increase $p_{0,j'}$ from zero to a non-zero value or decrease from a non-zero value to zero. The following is our algorithm:

Use the algorithm described in Theorem 6.1 to ensure that $p_{r,s} = q_{r,s}$ for all $r \leq n-j'$, $s \geq j'$, except possibly when $r = n-j'$, $s = j'$, **or** when $r = n-j'-1$, $s = j'$.
Use Lemma 3.1 on the coefficients $p_{n-j',j'}, p_{n-j',j'-1}, \ldots, p_{n-j',1}$ to ensure that each of these are non-zero.

If $p_{n-j',0} = 0$, then
  increase both $p_{n-j',1}$ and $p_{n-j'-1,1}$ by 1
  increase both $p_{n-j'-1,1}$ $p_{n-j'-1,1}$ and $p_{n-j'-1,0}$ by 1
(This will ensure that $p_{n-j'-1,1}, p_{n-j'-1,0} > 0$)
Similarly, also ensure that $p_{n-j'-2,1}, p_{n-j'-2,0} > 0$

For $t = 0, 1, 2, \ldots, j'-1$ do
  If $t > 0$ **or** $p_{n-j',0} > 0$ then
    modify the coefficients $p_{0,t}, p_{1,t}, \ldots, p_{n-j'-2,t}$, using Theorem 5.1, such that $p_{s,t} = q_{s,t}$ for $0 \leq s \leq n-j'-2$
  else
    modify the coefficients $p_{0,t}, p_{1,t}, \ldots, p_{n-j'-3,t}$, using Theorem 5.1, such that $p_{s,t} = q_{s,t}$ for $0 \leq s \leq n-j'-3$

  If $0 < t < j'-1$ then
    if $p_{n-j'-1,t} < q_{n-j'-1,t}$, then increase both $p_{n-j'-1,t}$ and $p_{n-j',t}$ by $q_{n-j'-1,t} - p_{n-j'-1,t}$
  if $p_{n-j',t} < q_{n-j',t}$, then increase both $p_{n-j',t}$ and $p_{n-j',t+1}$ by $q_{n-j',t} - p_{n-j',t}$
  if $p_{n-j',t} > q_{n-j',t}$ then
    increase both $p_{n-j',t+1}$ and $p_{n-j',t+2}$, if necessary, such that $p_{n-j',t+1} > p_{n-j',t} - q_{n-j',t}$
    decrease both $p_{n-j',t}$ and $p_{n-j',t+1}$ by $p_{n-j',t} - q_{n-j',t}$.
  If $p_{n-j'-1,t} \geq q_{n-j'-1,t}$ then
    increase both $p_{n-j',t}$ and $p_{n-j',t+1}$, if necessary, such that $p_{n-j',t} > p_{n-j'-1,t} - q_{n-j'-1,t}$
    decrease both $p_{n-j'-1,t}$ and $p_{n-j',t}$ by $p_{n-j'-1,t} - q_{n-j'-1,t}$
    if $p_{n-j',t} < q_{n-j',t}$ then
      increase both $p_{n-j',t}$ and $p_{n-j',t+1}$ by $q_{n-j',t} - p_{n-j',t}$
    if $p_{n-j',t} > q_{n-j',t}$ then
      increase both $p_{n-j',t+1}$ and $p_{n-j',t+2}$, if necessary, such that $p_{n-j',t+1} > p_{n-j',t} - q_{n-j',t}$
      decrease both $p_{n-j',t}$ and $p_{n-j',t+1}$ by $p_{n-j',t} - q_{n-j',t}$

  if $t = 0$ **and** $p_{n-j',0} > 0$ then
    if $p_{n-j',0} < q_{n-j',0}$, then increase both $p_{n-j'-1,0}$ and $p_{n-j',0}$ by $q_{n-j',0} - p_{n-j',0}$
    if $p_{n-j'-1,0} < q_{n-j'-1,0}$ then increase both $p_{n-j'-1,0}$ and $p_{n-j'-1,1}$ by $q_{n-j'-1,0} - p_{n-j'-1,0}$
    if $p_{n-j'-1,0} \geq q_{n-j'-1,0}$ then
      increase both $p_{n-j'-1,1}$ and $p_{n-j',1}$, if necessary, such that $p_{n-j'-1,1} > p_{n-j'-1,0} - q_{n-j'-1,0}$
      decrease both $p_{n-j'-1,0}$ and $p_{n-j'-1,1}$ by $p_{n-j'-1,0} - q_{n-j'-1,0}$

  if $t = 0$ **and** $p_{n-j',0} = 0$ then
    Apply the same procedure as for the case with $t = 0$ and $p_{n-j',0} > 0$, but replacing the quantity $n-j'$ by $n-j'-1$.
Endfor

Algorithm-A_1

It is easy to see that Algorithm-A_1 takes $O(n^2)$ time. At the end of Algorithm-A_1, we have changed P such that at most three of its coefficients, $p_{n-j'-1,j'}$, $p_{n-j',j'}$, $p_{n-j',j'-1}$, differ from the corresponding coefficients of Q. Now, we use Lemma 4.4 to determine in $O(1)$ time if $P =_\sigma Q$.

When $\ell > 1$:

For notational convenience, let $r_{\ell+1} = r_\ell$ and $s_{\ell+1} = 0$. First, again, use the proof of Theorem 6.1 to ensure that $p_{r,s} = q_{r,s}$ for all $r \leq n - j'$, $s \geq j'$, except possibly when $r = n - j', s = j'$, or when $r = n - j' - 1, s = j'$.

Now, we use Lemma 4.5 to process the convex vertices of $bd'(RCH(T))$ in the following way. In the k^{th} step of this (in this order $k = \ell, \ell - 1, \ldots, 1$), we modify the coefficients $p_{r,s}$, $r_k \leq r \leq r_{k+1}$, $s_{k+1} \leq s \leq s_k$, by essentially a similar procedure as described in Algorithm-A_1 such that they are the same as their corresponding coefficients in Q, except possibly the three coefficients p_{r_k,s_k}, p_{r_k-1,s_k} and p_{r_k,s_k-1}. Notice that since $p_{n-j',j'} > 0$, we can use Lemma 3.1 to ensure that $p_{n-j',s_k}, p_{n-j',s_k-1} > max\{|p_{r_k,s_k} - q_{r_k,s_k}|, |p_{r_k-1,s_k} - q_{r_k-1,s_k}|, |q_{r_k,s_k-1} - q_{r_k,s_k-1}|\}$. Hence, by Lemma 3.1 (parts (d) and (e)), we can correct the three coefficients, at the expense of modifying the coefficients p_{n-j',s_k} and p_{n-j',s_k-1}.

Finally, we apply Algorithm-A_1 to correct the coefficients $p_{r,s}$ for all $0 \leq r \leq n - j'$, $0 \leq s < j'$. The final result (unless we have already discovered that $P \neq_\sigma Q$) is a polynomial which differs from Q in at most three coefficients $p_{n-j',j'}, p_{n-j'-1,j'}, p_{n-j',j'-1}$. Now, we can use Lemma 4.4 to determine if $P =_\sigma Q$. ■

References

[1] E. D. Sontag, *Remarks on Piecewise-Linear Algebra*, Pacific Journal of Mathematics, Vol. 98, No. 1, pp. 183-201, 1982.