The Limit of Polynomials

Implications of Handelman's Theorem for Exploring Schedules

Tomofumi Yuki
Inria, Univ Rennes, CNRS, IRISA
tomofumi.yuki@inria.fr

Abstract

This paper studies the Handelman's theorem used for polynomial scheduling, which resembles the Farkas' lemma for affine scheduling. Theorems from real algebraic geometry and polynomial optimization show that some polynomials have Handelman representations when they are non-negative on a domain, instead of strictly positive as stated in Handelman's theorem. The global minimizers of a polynomial must be at the boundaries of the domain to have such a representation with finite bounds on the degree of monomials. This creates discrepancies in terms of polynomials included in the exploration space with a fixed bound on the monomial degree. Our findings give an explanation to our failed attempt to apply polynomial scheduling to Index-Set Splitting: we were precisely trying to find polynomials with global minimizers at the interior of a domain.

1 Introduction

The polyhedral model provides a concise and powerful abstraction of computations that fit the class of affine recurrences over polyhedral domains. The mathematical representation provides access to the rich set of linear algebraic properties, enabling powerful analyses and transformations.

One of the most frequently used property is the Affine form of Farkas' lemma that gives necessary and sufficient condition for an affine function to be positive on a polyhedron. The application of this lemma to characterize the space of legal schedules [3] is still the core engine for exploring the space of transformations in the polyhedral model.

The more general characterizations of the positive polynomials on semi-algebraic sets, so called Positivstellensatz, are an extensively studied topic in real algebraic geometry [16]. A theorem by Handelman [9] gives a characterization of positive polynomials on a polyhedron that resembles the Farkas' lemma. Feautrier proposed polynomial scheduling [5] based on this result.

The extension of the schedule space from affine functions to polynomials is exciting, but also raises an important question: when do we need polynomials? In fact, multi-dimensional affine schedules [3, 4] already include a chunk of polynomial schedules decomposed into multi-dimensional affine functions. We also found it challenging to find loop programs with polynomial controls or polynomial dependences. Although irregular code (e.g., with indirection arrays) is common, it is difficult to find regular but non-affine programs. There is also the problem of generating efficient code reflecting the polynomial schedules [6]. Moreover, the characterization of positive polynomials is similar but not a direct generalization of the Farkas' lemma. The practical implications of the differences are not fully clear.

In this paper, we first present an attempt to use polynomial scheduling for Index-Set Splitting (ISS) [8]. How to find the right split is the key question for ISS, and existing approaches find promising split planes by analyzing the dependences [1, 8]. Our attempt is based on the observation that some computations that do not have any valid affine schedules without ISS (or piece-wise affine schedules) have legal polynomial schedules. Our goal was to have a framework to reason about ISS that unifies existing heuristics through the additional expressive power of polynomial schedules.

Unfortunately, our attempt did not succeed due to various reasons, including those coming from inherent limitations of Handelman's theorem. We discuss the limitations of Handelman's theorem (and other related theorems on positive polynomials) with concrete examples that explain why we struggle to find a class of schedules in our approach for ISS. The main conclusion is that these theorems cannot be used to find a class of polynomials, which was exactly what we tried to find in our approach for ISS. The primary reason is that these theorems cannot *exactly* express polynomials that have global minimizers at the interior of the domain.

2 Background

In this paper, the readers are assumed to be familiar with the basic concepts of the polyhedral model [7].

2.1 Index Set Splitting

Index-Set Splitting is a transformation that aims at enlarging the legal schedule space of polyhedral computations [8]. Affine scheduling assigns an affine function as a schedule to each statement. However, there are cases where it is necessary to use two or more different schedules for a statement, depending on the specific instances of the statement. ISS can be applied as a pre-processing step to split the statements into multiple statements such that the scheduler can assign different scheduling functions.

ISS is a recurring concept with different motivations. For systolic array synthesis, ISS is used to localize affine dependences [21] where the objective is to only have uniform dependences in the scheduled space after split. Iteration space folding for tiling periodic stencils [1] share similarity with localization where the objective is to allow fully permutable schedules after split. The work by Griebl et al. [8] study cases where ISS gives better affine schedules in terms of latency.

Although these techniques have similarities, they greatly differ and are tailored for their respective context. Our original motivation was to use polynomial schedules as the common framework to unify these techniques.

2.2 Positivity Certificates

The core of the scheduling algorithms in the polyhedral model is the characterization of positive (non-negative) functions. In the following, we give a brief recap of the Farkas scheduling algorithm [3], Farkas' lemma, and Handelman's theorem for polynomial scheduling [5].

Given two statements S and T, and their respective scheduling functions f_S and f_T, a legal schedule must satisfy $f_{\mathsf{S}}(\vec{i}) > f_{\mathsf{T}}(\vec{i'})$ for all pairs of statement instances where $\mathsf{S}\langle\vec{i}\rangle$ depends on $\mathsf{T}\langle\vec{i'}\rangle$, expressed as a polyhedral set $[\vec{i},\vec{i'}] \in D$. This can be rewritten as $f_{ST}(\vec{i},\vec{i'}) > 0, [\vec{i},\vec{i'}] \in D$, where $f_{ST}(\vec{i},\vec{i'}) = f_S(\vec{i}) - f_T(\vec{i'})$. The problem of testing for legality of schedules can now be viewed as testing for positivity of affine functions within the domain D.

Then the Farkas' lemma (restated below) can be applied to characterize the space of legal affine schedules. The Farkas algorithm proceeds by exploring the solution space with ILP.

Theorem 1 (Affine From of Farkas' Lemma [3]). *Let $\mathcal{D}$ be a nonempty polyhedron defined by n affine inequalities:*

$$a_k.x + b_k \geq 0, \quad k = 1, n$$

Then an affine form ψ is nonnegative everywhere in $\mathcal{D}$ iff it is a positive affine combination:

$$\psi(x) \equiv \lambda_0 + \sum_k \lambda_k(a_k.x + b_k), \quad \lambda_k \geq 0$$

The polynomial scheduling is based on the same idea, but using the theorem by Handelman about positive polynomials on a polyhedron.

Theorem 2 (Handelman's Theorem [9]). *Let $\mathcal{D}$ be a compact nonempty polyhedron with interior defined by n affine inequalities (denoted as functions p):*

$$p_k(x) \geq 0, \quad k = 1, n$$

Then a polynomial f is strictly positive everywhere in $\mathcal{D}$ iff it is a positive linear combination of the monomials in $\mathcal{D}$:

$$f(x) \equiv \sum_{k \in \mathbb{N}^n} \lambda_k p_1^{k_1} \cdots p_n^{k_n}, \quad \lambda_k \geq 0 \tag{1}$$

where not all λs are zero.

Although it is similar to the Farkas' lemma, there are some important differences:

- It is characterizing *strictly positive* functions.
- The domain $\mathcal{D}$ must be compact, i.e., bounded.
- Most importantly, the positive combination is over monomials obtained as products of the constraints with unbounded degree. Thus, the application of this theorem requires a finite subset to be considered. Typically, the degree of the monomials are restricted by a parameter, called M, such that $|k| = \sum k_i \leq M$.

Note that there is a variant by Schweighofer [18] that allow arbitrary polynomials to be used as constraints on $\mathcal{D}$ provided that a subset of the constraints satisfy the same condition as Handelman. Since we only discuss computations with purely polyhedral domains and affine dependences, we use Handelman's theorem in this paper.

3 Polynomial Scheduling Guided ISS

In this section, we discuss our original motivation to use polynomial scheduling for index-set splitting. We first illustrate a possible use of polynomial scheduling for ISS with an example, and then describe a modified polynomial scheduling algorithm that focuses on finding simpler polynomials.

3.1 Simplified Floyd-Warshall

We use a simplified (2D version) of the Floyd-Warshall shortest paths algorithm to illustrate our idea. Floyd-Warshall requires ISS to be tiled using the commonly used sufficient condition for legal tiling (fully permutable). The computation before split can be tiled, but the tiles do not have a legal affine schedule, and prior work has used it as an example for dynamic scheduling of tiles [13].

The FW algorithm can be expressed as a single recurrence:

$$D(k,i,j) = min\left(\begin{array}{c} D(k-1,i,j), \\ \left(\begin{array}{c} k \geq (i,j) : D(k-1,i,k) + D(k-1,k,j) \\ j \leq k < i : D(k-1,i,k) + D(k,k,j) \\ i \leq k < j : D(k,i,k) + D(k-1,k,j) \\ k < (i,j) : D(k,i,k) + D(k,k,j) \end{array}\right) \end{array}\right)$$

defined over a cubic domain: $\{k,i,j : 1 \leq (i,j,k) \leq N\}$. The boundaries, $D(0,i,j)$, provide the initial values.

We simplify the above in two ways:

- The dimension is reduced to ease visualization. The important property is the dependence on the diagonal values, which can be preserved in 2D.
- The dependences across the k dimension are modified to always refer to $k-1$ to remove the cases. The different accesses becomes irrelevant after tiling.

This gives a much simpler recurrence:

$$X(k,i) = min\,(X(k-1,i), X(k-1,k-1))$$

and its dependences are illustrated in Figure 1.

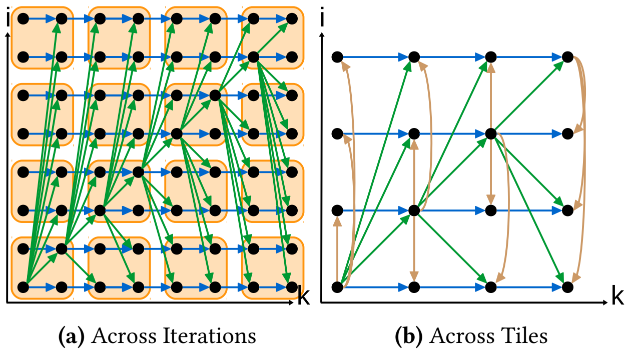

Figure 1. Dataflow dependences of the simplified FW. Figure 1a depicts the dependences across statement instances and Figure 1b show inter-tile dependences with 2×2 tiles.

3.2 Scheduling the Tiles

The original dependences do not satisfy fully permutability, but tiling is legal—inter-tile dependences are acyclic—when restricted to square tiles that are symmetric about the diagonal; the key property is that the sources of broadcast-like dataflow must be in a single tile for the broadcast dimensions (i and j). However, tiling introduces vertical inter-tile dependences that makes the resulting tiles to not have any legal affine schedule. For each column of tiles, the diagonal tile must execute first before other tiles, which is due to the broadcast-like dataflow from the diagonal.

There are multiple static schedules for these tiles. The following is an example of a possible piece-wise affine schedule:

$$\theta(k,i) = \begin{cases} i = k : (k, 0) \\ i \neq k : (k, 1) \end{cases}$$

It can also be given a polynomial schedules without pieces:

$$\theta(k,i) = kN^2 + (i-k)^2 \tag{2}$$

3.3 Polynomials to Splitting Planes

The key idea behind our approach is to use the expressiveness of the polynomials to extract information about profitable piece-wise decomposition of the original domain. Affine functions are uni-directional and are incapable of expressing schedules that increase in multiple directions, which is why the diagonals in FW pose problems.

This particular limitation does not apply to polynomials, e.g., quadratic functions increase in two directions. Thus, the main intuition is that the pieces necessary for piece-wise affine schedules may be found by analyzing the polynomials, scheduled without any ISS. For the polynomial $(i-k)^2$, the partial derivatives give the equality $i = k$ hinting the desired partitioning of the domain.

Our approach models the piece-wise decomposition in terms of hyperplanes that define the boundary of two pieces, called *splitting planes*. Given an affine hyperplane of the form $h(x) = 0$, the domain is decomposed into three pieces: $h(x) = 0$, $h(x) > 0$, and $h(x) < 0$. The objective is to find a set of splitting planes such that the pieces can be scheduled with affine functions.

We do not have a concrete algorithm for inferring splitting planes from polynomials. The basic steps of a potential algorithm are: (i) perform polynomial scheduling, (ii) analyze the polynomials to obtain splitting planes, and (iii) perform affine scheduling on the resulting piece-wise domain.

The analysis of polynomial schedules is the step that remains vague in our approach. The intuition is that if all the partial derivatives of the polynomial schedule do not change signs within a piece, then this piece should have a valid affine schedule. For the univariate case, this can be achieved by partitioning the domain into intervals defined by critical points of the polynomial. How the intuition generalizes to multivariate polynomials is unclear. A single critical point in a 2D space does not unambiguously define a splitting plane. For a polynomial in d dimensional space, the critical points must span a $d-1$ dimensional subspace to define a splitting plane. Moreover, critical points are overkill; the important property is that for each partition of the original domain, all partial derivatives do not change signs. The schedule (Eqn. 2) in the example above has no critical point due to the monomial kN^2—the partial derivative with respect to k is always positive, but the partial derivative with respect to i is zero on the line $i = k$.

Our original plan was to apply polynomial scheduling on a few examples to have a better understanding of what needs to be done in this step. As a first attempt, we seek for polynomials that have a specific format, $(a-b)^2$, which trivially exposes equalities forming a hyperplane. We consider it a success if we are able to find the schedule $(i-k)^2$ for the 2D case, and $(i-k)^2 + (j-k)^2$ for the 3D case.

3.4 Tweaking the Scheduling Algorithm

We take the polynomial scheduling algorithm by Feautrier [5] and make a few modifications to better suit our goal. We have no intention to use the resulting schedule as is, and are only interested in finding the splitting planes on computations that do not have legal affine schedules. Thus, the focus is on obtaining polynomial schedules that are easier to analyze and to extract splitting planes.

Multi-Dimensional Schedule The polynomial scheduling is used as an extension to the multi-dimensional Farkas algorithm [4]. The scheduler explores multi-dimensional affine schedules until the remaining dependences cannot be satisfied by any affine schedule. The scheduler then explores polynomial schedules for all remaining dependences.

Minimizing Number of Monomials The cost function for polynomial scheduler is set to minimize the number of monomials. The sum of absolute values of monomial coefficients is minimized. The monomials are weighted differently as a function of the monomial degree in the parameters.

3.5 Discussion

The FW example is obviously not the only situation where ISS is useful. The examples in the work by Griebl et al. [8] illustrate cases where the computations admit affine schedules, but ISS improves the latency. The attempt in this section is unlikely to be useful for ISS aiming at latency improvement, because affine schedules (with sub-optimal latency) exist.

The approach presented above also depends on heuristic reasoning over polynomial schedules found by the scheduler. One may say that it is merely shifting where the heuristic is applied to with respect to prior work on ISS. This is definitely true when the only example considered is FW—a simple heuristic to split the space by identifying the source of broadcast dependences is sufficient for handling this example. The original goal was to extend the approach to handle other uses of ISS, notably for periodic stencils [1], in a single framework, but we abandoned the idea since it does not even work for FW.

4 Results on Floyd-Warshall

In this section, we discuss the schedules found for FW examples with the modified polynomial scheduler. The scheduler was implemented with SAGEMath[1], which provides support for symbolic manipulation of polynomials and an interface for LP solvers (we used GLPK[2] as our backend).

4.1 Simplified Floyd-Warshall

The algorithm first finds an affine schedule, $\theta_0(k, i, N) = k$, to strongly satisfy all the dependences that span different values of k. The remaining dependences do not admit an affine schedule, and hence polynomial scheduling is used for the second dimension. We seek for a polynomial θ_1 that satisfies the following:

$$\theta_1(k, i, N) > \theta_1(k, k, N) : k > i \wedge D$$
$$\theta_1(k, i, N) > \theta_1(k, k, N) : k < i \wedge D$$

where $D = \{k, i | 1 \leq (k, i) \leq N\}$. This gives $\theta_1(k, i, N) = -2ki + i^2$ when $M = 2$; recall that M is the upper bound on the degree of monomials considered.

Observe that the value of θ_1 is negative when $k > i > 0$. This is because the above formulation did not include the positivity constraint. Adding the constraint $\theta_1(k, i, N) > 0$ in D gives $\theta_1(k, i, N) = N^2 - 2ki + i^2$. Note that the scheduler does not find $(k - i)^2$ even though it is a valid schedule with better cost since the monomial involving N has a higher cost than k^2. In fact, the polynomial $(k - i)^2$ cannot be expressed for the domain D using Handelman representation (Eqn. 1) when $M = 2$.

There are two possible ways to force the scheduler to find the desired schedule. The first one is to use a non-parametric domain, i.e., set N to some constant. The scheduler may be reformulated to test for existence of schedules of a certain form. When the schedule is constrained to take the form $(k - i)^2 + c$ where c is the constant that is minimized with LP, the scheduler finds $(k - i)^2 + 16$ when $M = 2$. This is $(k - i)^2 + N^2$, which is simply the earlier polynomial $N^2 - 2ki + i^2$ plus the forced monomial k^2. As the degree M is increased, the constant c can be reduced. For instance, the scheduler gives $(k - i)^2 + 3.2$ when $M = 8$. However, $(k - i)^2$ is not expressible with any M, although the constant continues to approach zero. The reasons are discussed in Section 5.

Another way to force the scheduler to find the desired schedule is to formulate the positivity constraint in a piece-wise manner. That is, if we replace the positivity constraint with the following:

$$\theta_1(k, i, N) > 0 : k > i \wedge D$$
$$\theta_1(k, i, N) > 0 : k < i \wedge D$$

then the scheduler have no problems finding $\theta_1(k, i, N) = (k - i)^2$ with $M = 2$. The reason why this works is also discussed in Section 5.

4.2 Floyd-Warshall

The scheduler behaves in a similar manner for the original Floyd-Warshall. The causality conditions are:

$$\theta_1(k, i, j, N) > \theta_1(k, k, j, N) : k \neq i \wedge D$$
$$\theta_1(k, i, j, N) > \theta_1(k, i, k, N) : k \neq j \wedge D$$

and the scheduler finds $i^2 + j^2 - 2ik - 2jk$ with $M = 2$ without the positivity constraint. Adding the positivity constraint gives $2N^2 + i^2 + j^2 - 2ik - 2jk$, and the scheduler can be forced to find the desired schedule, $(i - k)^2 + (j - k)^2$, by the same trick of making the positivity condition piece-wise as discussed above.

4.3 Discussion

The tweaks to the scheduling algorithm was effective in producing polynomials that are much easier to understand. It is interesting to note that a prototype scheduler by Feautrier [5] gave a schedule[3] with better latency, but with many monomials. The schedule achieved better latency by parallelizing across columns of tiles in Figure 1b, which only manifests when the number of tiles per column is six or more. Since we are not interested in this parallelism, the modified scheduler better suits our purpose.

However, the scheduler was unable to find the target schedules without modifying the problem formulation such that the positivity condition is given in a piece-wise manner. Inspecting the partial derivatives of the schedules does reveal the target hyperplanes, and hence they may be considered as acceptable results for these examples. Nonetheless, the

[1]http://www.sagemath.org/

[2]https://www.gnu.org/software/glpk/

[3]The schedule found by Feautrier's implementation for the simplified FW, which he kindly provided us through email exchanges, is $kN^2 + 2k^2N + 4kN + 3i^2 - k^3 - 2k^2 - 6ki - 2k$.

fact that the addition of positivity conditions prevents the scheduler from finding the desired schedule is alarming.

In fact, the piece-wise workaround is essentially exposing the $i = k$ boundary to the domains, which is the splitting plane that we are seeking for. The causality conditions already had the $i = k$ boundary to represent the broadcast dependence from diagonal tiles. Thus, the root of the problem is that the polynomial $(k-i)^2$ does not have a Handelman representation in $D = \{k, i | 1 \leq (k, i) \leq N\}$ with small M. Understanding why this is the case is the main subject of next section.

5 Understanding the Limits

Our attempt to apply polynomial scheduling to ISS led us to a case where a desired polynomial cannot be found with a reasonably small[4] bound on the degree of base monomials.

The main reason is the limitation of Handelman's theorem (or its ability to relax polynomial problems to LP) that there is no finite bound on M for certain polynomials. Lasserre [11] showed, in the context of polynomial optimization, that this is the case when a global minimizer of the polynomial is at the interior of the domain. Unfortunately, this is exactly the case when we seek to find splitting planes through polynomial scheduling: if the polynomial is minimized at the boundaries, then there is no domain to split.

5.1 LP Relaxation of Polynomial Optimization

Polynomial optimization is the minimization of a polynomial on a domain—polyhedron in our case. The problem definition (borrowed from Lasserre [11]) is to find p^* in the following:

$$p^* = \min_{x \in \mathbb{R}^d} \{g(x) | p_i(x) \geq 0, i = 1, n\} \tag{3}$$

where the feasible set $\{x \in \mathbb{R}^d | p_i(x) \geq 0, i = 1, n\}$ is denoted by $\mathcal{D}$. In this paper, we are interested in the case when $\mathcal{D}$ is a polyhedron, but p_i are polynomials in the original definition.

The LP relaxation applies Handelman's theorem (or other variants) to show the function

$$g(x) - p^* + \epsilon \tag{4}$$

is strictly positive on the domain in question (ϵ is an arbitrary small number so that the function is strictly positive). The main intuition is that the Handelman representation with sufficiently high M can represent Eqn. 4. The lower bound steadily improves as M increases and eventually converges to the exact solution [11, 20].

The LP relaxation is not using the Handelman's theorem per se; it was developed separately and the connection was later established. The LP relaxation uses products of constraints (the exact same ones as in Handelman's theorem) where each product gives an additional constraint since the products are all non-negative. All the monomials obtained through products of constraints are replaced with new variables such that the target polynomial is expressed as a linear function of the new variables. This results in some loss of information; for instance, the monomials x and x^2 are replaced with new variables, a and b, without expressing the relation $a^2 = b$. The intuition is that the products of constraints adds more and more (valid) constraints to these variables as M increases, eventually constraining the variables to respect polynomial relations that cannot be encoded.

[4]For $(k - i)^2$ in $\{k, i | 1 \leq (k, i) \leq N\}$, we tried up to $M = 16$.

5.2 Scheduling View vs Minimization View

We take Example 2.4 by Lasserre [11] to illustrate the differences between the two uses of Handelman's theorem. In the following, we say *scheduling-view* to refer to the use of Handelman's theorem in polynomial scheduling [5], although it is applied to the minimization problem.

We are interested in minimizing a polynomial $x^2 - x$ for $x \in [0, 1]$. Thus, we have:

- $g(x) = x^2 - x$,
- $p_0 = x$, and
- $p_1 = 1 - x$.

The solution p^* is $-\frac{1}{4}$.

With $M = 2$, all possible products of the constraints up to degree 2 are considered:

$$[1, x, 1 - x, x^2, x(1 - x), (1 - x)^2]$$

and with $M = 3$ the following constraints are considered in addition to those for $M = 2$:

$$[x^3, x^2(1 - x), x(1 - x)^2, (1 - x)^3]$$

Scheduling View

The scheduling-view considers the positive combination of the products by introducing λ coefficients:

$$\lambda_0 + \lambda_1 x + \lambda_2(1 - x) + \lambda_3 x^2 + \lambda_4 x(1 - x) + \lambda_5(1 - x)^2$$

factorizes the expression by the monomials:

$$\begin{aligned} &(\lambda_0 + \lambda_2 + \lambda_5) \times 1 \\ &+(\lambda_1 - \lambda_2 + \lambda_4 - 2\lambda_5) \times x \\ &+(\lambda_3 - \lambda_4 + \lambda_5) \times x^2 \end{aligned}$$

and solves the following LP problem:

$$\begin{array}{rlcr} \max & -(\lambda_0 + \lambda_2 + \lambda_5) & & \\ \text{such that} & \lambda_1 - \lambda_2 + \lambda_4 - 2\lambda_5 & = & -1 \\ & \lambda_3 - \lambda_4 + \lambda_5 & = & 1 \end{array}$$

where the polynomial is forced to be $x^2 - x$, and the objective gives the lower bound on p^*.

The solution with $M = 2$ is $x^2 - x > -\frac{1}{2}$ when $\lambda_3 = \lambda_5 = \frac{1}{2}$. With $M = 3$, it becomes $-\frac{1}{3}$; $\frac{1}{3}(x^3 + (1-x)^3) = x^2 - x + \frac{1}{3} > 0$. Higher values of M enable new possibilities to express the target polynomial with smaller constant.

Minimization View

The minimization-view takes the products of constraints:

$$[1, x, 1-x, x^2, x-x^2, x^2-2x+1]$$

replaces the monomials with variables ($x = y_1$ and $x^2 = y_2$):

$$\vec{c} = [1, y_1, 1-y_1, y_2, y_1-y_2, y_2-2y_1+1]$$

and solves the following LP problem:

$$\begin{aligned} \min \;& y_2 - y_1 \\ \text{such that } \;& \vec{c} \geq \vec{0} \end{aligned}$$

where the objective function directly expresses the polynomial in terms of new variables.

The solution with $M = 2$ is $\frac{1}{2}$ when $y_1 = \frac{1}{2}$ and $y_2 = 0$. With $M = 3$, the additional constraints on new variables (including those for $x^3 = y_3$) are added:

$$[y_3, y_2 - y_3, y_3 - 2y_2 + y_1, -y_3 + 3y_2 - 3y_1 + 1] \geq \vec{0}$$

making $y_1 = \frac{1}{2}$ no longer feasible due to the last constraint. The new solution is $-\frac{1}{3}$ when $y_1 = \frac{1}{3}$ and $y_2 = y_3 = 0$. Additional constraints on y are added as higher values of M are considered, making the solution converge towards p^*.

Duality

The scheduling-view is a more direct application of Handelman's theorem, and it strictly looks at valid Handelman representations. The minimization-view can be explained by Handelman's theorem, but is focused on the minimal value of a polynomial. In fact, the two views are duals of each other. The scheduling-view is an LP where the variables are the coefficients of the positive combination in Handelman representation, and the LP constraints corresponds to monomials. The minimization-view is the other way around.

5.3 Exactness of the Relaxation

The main result by Lasserre [11] (restated below) we borrow is Theorem 3.1 that state that the Handelman representation cannot give exact solution to the minimization problem when a global minimizer is at the interior of the domain. In other words, Handelman representation with $\epsilon = 0$ (recall Eqn. 4) exists only if the global minimizers of the polynomial are at the boundaries of the domain.

Theorem 3 (Theorem 3.1 by Lasserre [11])**.** *Consider the problem in Eqn. 4 and the LP relaxation of the problem with the degree of monomials bounded by M. Let p_M be the optimal value of the relaxed problem. Then,*

(a) For every M, $p_M \leq p^*$ and

$$g(x) - p_M = \sum_{|k| \leq M} \lambda_k p_1(x)^{k_1} \cdots p_n(x)^{k_n}, x \in \mathbb{R}^d \tag{5}$$

for some non-negative scalars $\{\lambda_k\}$. Let x^* be a global minimizer of the problem in Eqn. 4 and let $I(x^*)$ be the set of active constraints at x^*. If $I(x^*) = \emptyset$ (i.e., x^* is in the interior of $\mathcal{D}$) or if there is some feasible, nonoptimal solution $x \in \mathcal{D}$ with $p_i(x) = 0, \forall i \in I(x^*)$, then $p_M < p^*$ for all M, that is, no relaxation can be exact.

(b) If all the p_i are linear, that is, if $\mathcal{D}$ is a convex polytope, then Eqn. 5 holds and $p_M \uparrow p^*$ as $M \to \infty$. If $I(x^*) = \emptyset$ for some global minimzer x^*, then in Eqn. 5,

$$\sum_k \lambda_k \to 0 \text{ as } M \to \infty \tag{6}$$

The above is given in the polynomial optimization context, but the results are equally applicable to the scheduling context through the duality (the dual problem is used as part of Theorem 3.1).

5.4 Relation to Strict Positivity

The result by Lasserre [11] has direct links to the strict positivity in Handelman's theorem. In fact, the cases where the exact solution to the minimization problem cannot be reached is exactly where strict positivity is required. In other words, Handelman representation can express polynomials that are non-negative on a polyhedron if the global minimizers of a polynomial are not at the interior.

Observe that in Handelman's theorem (Eqn. 1) the target polynomial is represented as a linear combination of products of constraints where each constraint is non-negative. Intuitively, such a representation should be able to describe non-negative polynomials in some cases. It is easy to find examples where a polynomial that is non-negative in a polyhedron has a corresponding Handelman representation. For instance, x^2 for $x \in [0, 1]$ gives a trivial example. There are also many examples where non-negative polynomials cannot be expressed with Handelman's theorem. A simple example is x^2 for $x \in [-1, 1]$. Lasserre's result gives a clear characterization of when non-negative polynomials can be represented by Handelman's theorem.

Note that showing x^2 is non-negative for $x \in [-1, 1]$ is trivial once the domain is split into two: $[-1, 0]$ and $[0, 1]$. This is essentially the same "workaround" as the piece-wise specification of positivity constraint. The global minimizers are now at the boundaries introduced by the piece-wise split.

5.5 Constant Matters

The minimization-view of Handelman's theorem directly reveals an implication for scheduling: the constant matters. Given a polynomial $f(x)$, the difficulty of expressing $f(x)+b$ with Handelman representation increases as b shrinks.

For instance, expressing $x^2 + 1$ for $x \in [-1, 1]$ requires $M = 2$, but $x^2 + 0.5$ on the same domain requires $M = 3$, and $x^2 + 0.25$ requires $M = 5$. Enlarging the domain also has the same effect: $x^2 + 1$ for $x \in [-2, 2]$ requires $M = 5$.

Known bounds on the degree for other variants of Positivstellensatz include a metric of how close the polynomial is to zero as an important component, along with degree of the polynomial and dimensionality of the space [14, 19].

5.6 Parametric Domains

One important difference between Farkas's lemma and Handelman's theorem is that Handleman's theorem is limited to convex polytopes. In other words, parametric polyhedra, which are frequently used in polyhedral compilation, are technically not satisfying an assumption of the theorem. However, experiences with polynomial scheduling show that some polynomials have Handelman representations even when the domain is parametric. The results by Lasserre [11] also shed some light to the treatment of parametric domains.

As briefly mentioned during the discussion of constants, expressing $x^2 + 0.25$ for $x \in [-1, 1]$ is no different from expressing $x^2 + 1$ for $x \in [-2, 2]$, since one is simply a scaled version of another. This also means that the difficulty of finding a Handelman representation for a polynomial increases as the domain enlarges. The lack of exact solution to the minimization problem also implies that there is no Handelman representation for a parametric polyhedron when the polynomial has global minimizers at the interior. (For larger domains, infinitely higher degree monomials are required.)

We use an example where we seek $x^2 + 1$ for $x \in [-N, N]$ to illustrate what happens. We have two constraints:

$$[x + N, N - x]$$

when $N = 1$ the Handelman representation can be found with $M = 2$: $\frac{1}{2}(x+1)^2 + \frac{1}{2}(1-x)^2$. An alternative view of the above, which may be more friendly for humans, is that a polynomial $b(x^2 + 1)$ where b is a constant is first constructed with some combination of the integer multiples of the constraint products. In the above, $(x+1)^2 + (1-x)^2 = 2x^2 + 2$, and dividing all the coefficients by two gives the desired polynomial. However, it becomes more and more complicated with larger values of N. The solution when $N = 2$ is:

$$\frac{1}{1024}\begin{pmatrix} 32(x+2)^3+ \\ 8(x+2)(2-x)^3+ \\ 3(x+2)^5+ \\ 5(x+2)^4(2-x)+ \\ 7(x+2)(2-x)^4+ \\ 5(2-x)^5 \end{pmatrix}$$

which is already much more complicated, and it only gets worse for larger values of N. As stated in the latter half of Theorem 3.1 [11], the λ coefficients converge toward zero.

5.7 Implications for Scheduling

The main implication is that certain families of polynomials are not expressible by Handelman's theorem, restricted to some fixed bound on the constraint products. Additionally, it is important to keep in mind that constants play a major role in determining the difficulty of expressing a polynomial. This may lead to unintended consequences when scheduling multiple statements where one should have a constant offset with respect to another.

The polynomials with global minimizers at the interior of a domain are essentially excluded from candidate schedules, especially when the domain is parametric. This turned out to be a major issue for our original goal of finding splitting planes. How restrictive this limitation is for other uses of Handelman's theorem (and its variants) are not obvious. In particular, those concerned on existence of a polynomial schedule [2, 15] rather than finding a "good" schedule is unlikely to be affected.

Moreover, for polynomials with global minimizers at the boundaries, we can obtain non-negativity certificates with Handelman representations. Since polynomials with global minimizers at the interior are excluded from the solution space, the strict positivity could be practically ignored, and the scheduler may assume non-negativity.

6 Related Work and Discussion

In this section, we discuss other forms of positivity certificates and place our observations regarding Handelman's theorem into context.

6.1 Schweighofer's Theorem

Schweighofer's theorem [18] is a generalization of Handelman's theorem that allows domains to be partially defined by polynomial constraints. The results by Lasserre [11] are slightly different when the domain is a general semi-algebraic set: LP-relaxation converges to the optimal solution when $M \to \infty$ for convex polytopes, but this does not hold for general semi-algebraic sets. This difference is not of significant importance for the purpose of this paper; it does not change the fact that polynomials with global minimizers at the interior cannot be expressed with some finite M.

6.2 Sum of Squares and SDP-Relaxation

There is a large body of work on positivity certificates of polynomials using sum of squares representation. An article by Scheiderer [16] gives an overview of this subject, including non-negativity certificates.

These alternate forms of positivity certificates can be used to solve the polynomial optimization problem through relaxations to Semi-Definite Programming [10, 12]. The basic procedure is similar to the LP-relaxation; the bounds on the degree of monomials under consideration must be fixed. The work by Lasserre [10] compares LP and SDP-relaxations with the conclusion that SDP is better suited to the problem. The various non-negativity certificates [16] as well as bounds on the degree [14, 19] are also indirect evidences.

It would be interesting to see if the SDP-relaxation can be used as an alternative approach to polynomial scheduling. Sum of squares representation may allow for non-negativity certificates for convex polytopes, which would remove one of the main differences with Farkas' lemma. However, recent developments on Positivstellensatz essentially all rely on the domain to be compact, which was the important assumption in Schmüdgen's theorem [17] distinguishing it from earlier ones. Thus, the problem with parametric domain still remains. Note that the discussion in Section 5.6 only gives an explanation for why Handelman representations do not exist for polynomials with the global minimizers at the interior of a parametric polyhedron; if the converse holds is still open.

7 Conclusion

In this paper, we describe a potential application of polynomial scheduling to index-set splitting. Although ISS seems like an interesting use of the additional expressive power given by polynomials, we failed to obtain expected results.

Further study of Handelman's theorem and related work on positivity certificates gives some insights about the behavior of our polynomial scheduler. The theorem by Lasserre [11] explains why Handelman's theorem characterize strictly positive polynomials, and describe a class of polynomials that cannot be exactly expressed. The unfortunate conclusion is that the kind of schedules we seek to find in the context of ISS is not expressible with polynomial scheduling based on Handelman's theorem.

It appears that approaches based on SDP is the clear winner (with respect to LP) when it comes to relaxation of polynomial optimization problems. An interesting future work is to apply the ideas from SDP-relaxation to scheduling, which may finally provide a new approach for index-set splitting.

Acknowledgement

We would like to thank Paul Feautrier for discussions about polynomials and his help for implementing a polynomial scheduler. We thank Adrien Chaffangeon, Adrien Gougeon, and Anne Timothee for their work on trying polynomial scheduler on ISS, and implementation of a prototype scheduler. We also thank Steven Derrien and Silviu-Ioan Filip for various discussions.

References

[1] Uday Bondhugula, Vinayaka Bandishti, Albert Cohen, Guillain Potron, and Nicolas Vasilache. 2014. Tiling and optimizing time-iterated computations over periodic domains. In *Proceedings of the 23rd International Conference on Parallel Architecture and Compilation Techniques (PACT '14)*. 39–50.

[2] Albert Cohen, Alain Darte, and Paul Feautrier. 2016. Static Analysis of OpenStream Programs. In *Proceedings of the 6th International Workshop on Polyhedral Compilation Techniques (IMPACT '16)*.

[3] Paul Feautrier. 1992. Some Efficient Solutions to the Affine Scheduling Problem, I, One-dimensional Time. *International Journal of Parallel Programming* 21, 5 (1992), 313–348.

[4] Paul Feautrier. 1992. Some Efficient Solutions to the Affine Scheduling Problem, II, Multidimensional Time. *International Journal of Parallel Programming* 21, 6 (1992), 389–420.

[5] Paul Feautrier. 2015. The Power of Polynomials. In *Proceedings of the 5th International Workshop on Polyhedral Compilation Techniques (IMPACT '15)*.

[6] Paul Feautrier and Albert Cohen. 2018. On Polynomial Code Generation. In *Proceedings of the 8th International Workshop on Polyhedral Compilation Techniques (IMPACT '18)*.

[7] Paul Feautrier and Christian Lengauer. 2011. The Polyhedron Model. In *Encyclopedia of Parallel Programming*, David Padua (Ed.). Springer.

[8] Martin Griebl, Paul Feautrier, and Christian Lengauer. 2000. Index Set Splitting. *International Journal of Parallel Programming* 28, 6 (Dec 2000), 607–631.

[9] David Handelman. 1988. Representing polynomials by positive linear functions on compact convex polyhedra. *Pacific J. Math.* 132, 1 (1988), 35–62.

[10] J. Lasserre. 2001. Global Optimization with Polynomials and the Problem of Moments. *SIAM Journal on Optimization* 11, 3 (2001), 796–817.

[11] Jean B. Lasserre. 2002. Semidefinite Programming vs. LP Relaxations for Polynomial Programming. *Mathematics of Operations Research* 27, 2 (May 2002), 347–360.

[12] Monique Laurent. 2009. *Sums of Squares, Moment Matrices and Optimization Over Polynomials*. Springer, 157–270.

[13] Ravi Teja Mullapudi and Uday Bondhugula. 2014. Tiling for Dynamic Scheduling. In *Proceedings of the 4th International Workshop on Polyhedral Compilation Techniques (IMPACT '14)*.

[14] Jiawang Nie and Markus Schweighofer. 2007. On the complexity of Putinar's Positivstellensatz. *Journal of Complexity* 23, 1 (2007), 135 – 150.

[15] Diogo N. Sampaio, Louis-Noël Pouchet, and Fabrice Rastello. 2017. Simplification and Runtime Resolution of Data Dependence Constraints for Loop Transformations. In *Proceedings of the International Conference on Supercomputing (ICS '17)*. ACM, Article 10, 11 pages.

[16] Claus Scheiderer. 2009. *Positivity and Sums of Squares: A Guide to Recent Results*. Springer, 271–324.

[17] Konrad Schmüdgen. 1991. The K-moment problem for compact semi-algebraic sets. *Math. Ann.* 289, 1 (01 Mar 1991), 203–206.

[18] Markus Schweighofer. 2002. An algorithmic approach to Schmüdgen's Positivstellensatz. *Journal of Pure and Applied Algebra* 166, 3 (2002), 307–319.

[19] Markus Schweighofer. 2004. On the complexity of Schmüdgen's Positivstellensatz. *Journal of Complexity* 20, 4 (2004), 529–543.

[20] Hanif D. Sherali and Cihan H. Tuncbilek. 1992. A global optimization algorithm for polynomial programming problems using a Reformulation-Linearization Technique. *Journal of Global Optimization* 2, 1 (01 Mar 1992), 101–112.

[21] Yoav Yaacoby and Peter R. Cappello. 1988. Converting affine recurrence equations to quasi-uniform recurrence equations. In *VLSI Algorithms and Architectures*, John H. Reif (Ed.). 319–328.