Counting-based Loop Optimization

Philippe Clauss
University of Strasbourg + Inria
France

Integer points counting

Exact number of integer points inside a parametrized polytope:

- *polytope*: finite polyhedron

  iteration domain of a loop nest, *front of parallel iterations*, ...

- *parameterized* polytope

  = polytope linearly dependent on integer parameters $n, m, \ldots$

  = defined by constraints of the form $ai + bj + \ldots + \alpha n + \beta m + \ldots \geq 0$

Integer points counting

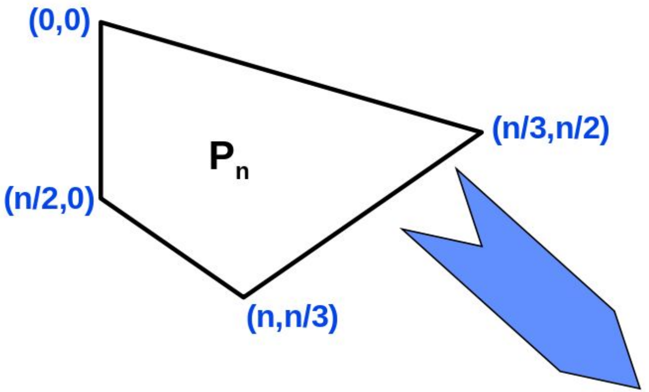
Exact number of integer points inside a parametrized polytope :

- A set of *Ehrhart polynomials* (also called *quasi-polynomials*)
- defined on adjacent convex domains (or chambers) of the parameters values

Ehrhart polynomial: a (kind of) multivariate polynomial

- whose variables are the parameters
- whose degree is the dimension of the polytope
- whose coefficients are *periodic numbers*
  - period = *lcm* of the vertices coordinates denominators

Ehrhart polynomial



$$period = lcm\,(0,2,3) = 6$$

$$n\,modulo\,6 = 0:\ \frac{5}{18}n^2 + \frac{1}{2}n + 1$$

$$n\,modulo\,6 = 1:\ \frac{5}{18}n^2 + \frac{4}{9}n + \frac{5}{18}$$

$$n\,modulo\,6 = 2:\ \frac{5}{18}n^2 + \frac{7}{18}n + \frac{1}{9}$$

$$\ldots$$

$$\tfrac{5}{18}n^2 + [\tfrac{1}{2}, \tfrac{4}{9}, \tfrac{7}{18}]n + [1, \tfrac{5}{18}, \tfrac{1}{9}, 1, -\tfrac{2}{9}, \tfrac{1}{9}]$$

- Ph. Clauss. *Counting Solutions to Linear and Nonlinear Constraints Through Ehrhart Polynomials: Applications to Analyze and Transform Scientific Programs*. 10th International Conference on Supercomputing, ICS'96, 1996.
- Ph. Clauss and V. Loechner. *Parametric Analysis of Polyhedral Iteration Spaces*. Journal of VLSI Signal Processing, 1998.

Barvinok

- Another approach for parametric counting
- Based on mathematician Barvinok's results
- More robust than the previous interpolation-based method
- Periodic numbers ⇒ floors & ceilings
- Implemented in the barvinok library (iscc)

```
card [n]->{[i,j]: 0<=2*i<n and 3*i<=3*j<n};
$0 := [n] -> { (1/2 * floor((2 + n)/3) + 1/2 * floor((2 + n)/3)^2) : n > 0 }
```


- Sven Verdoolaege, Rachid Seghir, Kristof Beyls, Vincent Loechner, Maurice Bruynooghe: Counting Integer Points in Parametric Polytopes Using Barvinok's Rational Functions. Algorithmica 48(1): 37-66 (2007)

Counting-based optimizations

Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Counting-based optimizations

Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Counting-based optimizations

Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Counting-based optimizations

Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Counting-based optimizations

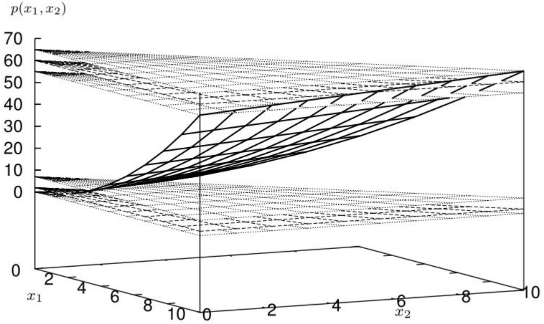
Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Polynomial maximization



- Bernstein polynomials:
  - Form a basis for the space of polynomials
  - Any polynomial can be expressed in this basis through *Bernstein coefficients*
  - The value of the polynomial is bounded by the values of the *minimum and maximum Bernstein coefficients*
  - The direct formula allows *symbolic computation* of the Bernstein coefficients
  - Extension of Bernstein expansion for parameterized multivariate polynomial expressions defined over *parametric convex polytopes*

- Ph. Clauss and I. Tchoupaeva. *A Symbolic Approach to Bernstein Expansion for Program Analysis and Optimization*. In 13th International Conference on Compiler Construction, CC 2004.
- Ph. Clauss, F. J. Fernández, D. Garbervetsky, and S. Verdoolaege. *Symbolic polynomial maximization over convex sets and its application to memory requirement estimation*. IEEE Trans. on Very Large Scale Integration (VLSI) Systems, 2009.

Bernstein expansion on parametric polytopes

- Implemented in the barvinok library (*iscc*)

```
ub [N]->{[x,y] -> 1/2*x^2+1/2*x+3/2*y^2-y : 0<=x<=N and x<=y<=N};

$0 := ([N] -> { max((-1/2 * N + 2 * N^2)) : N >= 0 }, True)

lb [N]->{[x,y] -> 1/2*x^2+1/2*x+3/2*y^2-y : 0<=x<=N and x<=y<=N};

$1 := ([N] -> { min(-1/2 * N) : N >= 0 }, False)
```

$$\forall\ (x,y)\ s.\,t.\ 0 \leq x \leq N\, and\, x \leq y \leq N,$$
$$-\frac{1}{2}N \leq \frac{1}{2}x^2 + \frac{1}{2}x + \frac{3}{2}y^2 - y \leq 2N^2 - \frac{1}{2}N$$

Counting-based optimizations

Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Memory requirement analysis

- For a given temporary array
  - Liveness analysis
    - Number of live elements at any iteration I
    - Live(I) = #reads after I - #associated last writes after I

      = Ehrhart polynomial
  - ub Live(I)

    = maximum number of live elements

    = size of the contracted temporary array

- Ph. Clauss, D. Garbervetsky, V. Loechner, and S. Verdoolaege. *Polyhedral Techniques for Parametric Memory Requirement Estimation*. In F. Balasa and D. Pradhan, editors, Energy-Aware Memory Management for Embedded Multimedia Systems: A Computer-Aided Design Approach, Chapman & Hall/Crc Computer and Information Science. Taylor and Francis, 2011.

Counting-based optimizations

Loop optimization

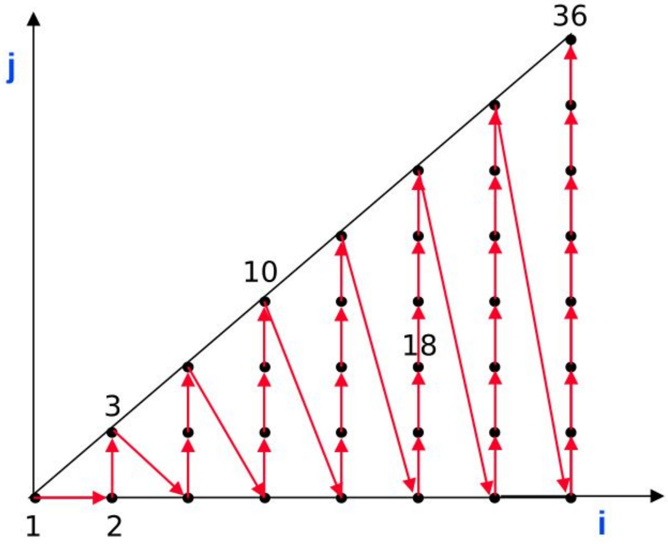
- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Ranking polynomials

- Rank (or position) of an integer point
  - inside a polytope
  - whose points are listed in lexicographical order



$$(i', j') \leq_{lex} (i, j) \implies \left\{ \begin{matrix} i' < i \\ or \\ i' = i \, and \, j' \leq j \end{matrix} \right\}$$

$$Rank\,(i, j) = \#\left\{ (i', j')\; s.\,t.\; (i', j') \leq_{lex} (i, j) \right\}$$

$$= \left\{ \begin{matrix} \#\{(i', j')\; s.\,t.\; i' < i\} \\ + \\ \#\{(i', j')\; s.\,t.\; i' = i,\; j' \leq j\} \end{matrix} \right\}$$

Ranking polynomials

- Example:

$$\mathbb{P} = \{(i,j,k) \in \mathbb{Z}^3 \mid 0 \leq i < N, 0 \leq j \leq i, 0 \leq k < M\}$$

$$R(i,j,k) \quad = \#\left\{(i',j',k') \in \mathbb{P} \,\middle|\, \begin{array}{l} (i,j,k) \in \mathbb{P} \\ (i',j',k') \leq_{lex} (i,j,k) \end{array}\right\}$$

$$\begin{aligned} &= \#\{(i',j',k') \in \mathbb{P} \mid (i,j,k) \in \mathbb{P}, i' < i\} \\ &+ \#\{(i',j',k') \in \mathbb{P} \mid (i,j,k) \in \mathbb{P}, i' = i, j' < j\} \\ &+ \#\{(i',j',k') \in \mathbb{P} \mid (i,j,k) \in \mathbb{P}, i' = i, j' = j, k' \leq k\} \end{aligned}$$

$$= \boxed{(2\,k + 2\,M\,j + M\,i^2 + M\,i + 2)/2}$$

Counting-based optimizations

Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Data layout transformation

- Goal: Spatial data locality
  - Organize array elements in the order in which they are accessed
  - Assign to array elements the rank of the iterations referencing them
    - Each array element A[X] is referenced at iterations I s.t. X=f(I), where f is the affine array reference function
    - If A[X] is referenced more than once, select I_{min} = lexmin(I)
    - $\#\{I'_{min} \leq_{lex} I_{min}\}$ is the new index of A[X]

- Ph. Clauss and B. Meister. *Automatic memory layout transformation to optimize spatial locality in parameterized loop nests*. ACM SIGARCH, Computer Architecture News, 28(1), march 2000.
- Vincent Loechner, Benoit Meister, and Philippe Clauss. *Precise data locality optimization of nested loops*. Journal of Supercomputing, 21(1):3776, January 2002.

Counting-based optimizations

Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Trahrhe expressions

- Opposite problem to ranking: unranking
- Given a point's rank of an integer point in *D*, what are its integer coordinates?

⇒ ranking polynomial inversion:

- Let p be a rank and $R(I)$ be a ranking polynomial: Find I s.t. $R(I) = p$
- Find the reverse function $R^{-1}(p) = T(p)$
- $T(p)$ is a sequence of algebraic expressions

$$t_1(p),\ t_2(p),\ t_3(p),\ \ldots,t_d(p)$$

TRAHRHE

Trahrhe expressions

- Example:
  - $\mathbb{P} = \{(i,j,k) \in \mathbb{Z}^3 \mid 0 \le i < N, 0 \le j \le i, 0 \le k < M\}$
    $R(i,j,k) = (2\,k + 2\,M\,j + M\,i^2 + M\,i + 2)/2$

1. Roots of $R(i,0,0) - p = (M\,i^2 + M\,i + 2)/2 - p$

- 2 roots : $r_1(p) = -\frac{\sqrt{8\,M\,p + M^2 - 8\,M} + M}{2\,M}$, $r_2(p) = \frac{\sqrt{8\,M\,p + M^2 - 8\,M} - M}{2\,M}$
- Minimum value of $i = 0$
- $r_2(1) = 0 \Rightarrow$ select and propagate :

$$\boxed{t_1(p) = \left\lfloor \frac{\sqrt{8\,M\,p + M^2 - 8\,M} - M}{2\,M} \right\rfloor}$$

Trahrhe expressions

- Example (continued):

  2. Root of
  $R(t_1(p), j, 0) - p = (2\,M\,j + M\,t_1(p)^2 + M\,t_1(p) + 2)/2 - p$

     ► Propagate $\boxed{t_2(p) = \left\lfloor -\frac{M\,t_1(p)^2 + M\,t_1(p) - 2\,p + 2}{2\,M} \right\rfloor}$

  3. $$\begin{aligned} t_3(p) &= p - R(t_1(p), t_2(p), 0) \\ &= \boxed{-\frac{2\,M\,t_2(p) + M\,t_1(p)^2 + M\,t_1(p) - 2\,p + 2}{2}} \end{aligned}$$

Trahrhe expressions

- Example (continued):
  - $R(0,0,0)=1,\ R(0,0,1)=2,\ R(0,0,2)=3,\ \ldots,$
    $R(0,0,M-1)=M,\ \ldots,\ R(1,0,0)=M+1,\ \ldots,$
    $R(N-1,N-1,M-1)=M\,N\,(N+1)/2$

    $T(p)=(t_1(p),t_2(p),t_3(p))=$
    $\left(\left\lfloor\frac{\sqrt{8\,M\,p+M^2-8\,M}-M}{2\,M}\right\rfloor, \left\lfloor-\frac{M\,t_1(p)^2+M\,t_1(p)-2\,p+2}{2\,M}\right\rfloor, -\frac{2\,M\,t_2(p)+M\,t_1(p)^2+M\,t_1(p)-2\,p+2}{2}\right)$
  - $T(1)=(0,0,0)$
  - $T(2)=\left(\left\lfloor\frac{\sqrt{M\,(M+8)}-M}{2\,M}\right\rfloor, \left\lfloor\frac{1}{M}\right\rfloor, \frac{2}{2}\right)=(0,0,1)$
  - $T(M)=\left(\left\lfloor\frac{\sqrt{M\,(9\,M-8)}-M}{2\,M}\right\rfloor, \left\lfloor\frac{M-1}{M}\right\rfloor, \frac{2\,M-2}{2}\right)=(0,0,M-1)$
  - ...

Counting-based optimizations

Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Collapsing of non-rectangular loops

- loop collapsing:

$$
\begin{aligned}
&for\ (i = 0;\ i < N;\ i++)\\
&\quad for\ (j = 0;\ j < M;\ j++)\\
&\quad\quad \{\dots\}
\end{aligned}
$$

$$\Downarrow$$

$$
\begin{aligned}
&for\ (pc = 0;\ pc < N * M;\ pc++)\\
&\quad \{\dots\}
\end{aligned}
$$

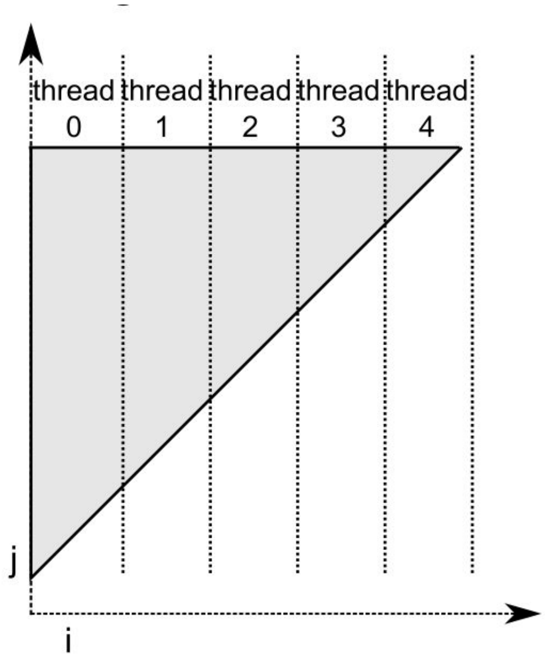
- Implemented in OpenMP (clause `collapse(n)`):
  - OpenMP 3.0: only constant loop bounds
  - OpenMP 5.0: bounds depending only on one unique loop index (otherwise : rectangular hull of the loop nest)

Collapsing of non-rectangular loops

- Non-rectangular loops: load imbalance issue

  Example: correlation kernel using 5 threads

$$
\begin{array}{l}
\#\text{pragma omp parallel for private(j,k)}\\
for\ (i = 0;\ i < N - 1;\ i++)\\
\ for\ (j = {\color{red} i + 1};\ j < N; j++)\ \{\\
\ \ for\ (k = 0;\ k < N;\ k++)\\
\ \ \ \ a\,[i]\,[j] += b\,[k]\,[i] * c\,[k]\,[j];\\
\ \ \ a\,[j]\,[i] = a\,[i]\,[j];\ \}
\end{array}
$$



- `omp schedule(dynamic)`:
  - time overhead + scalability issues

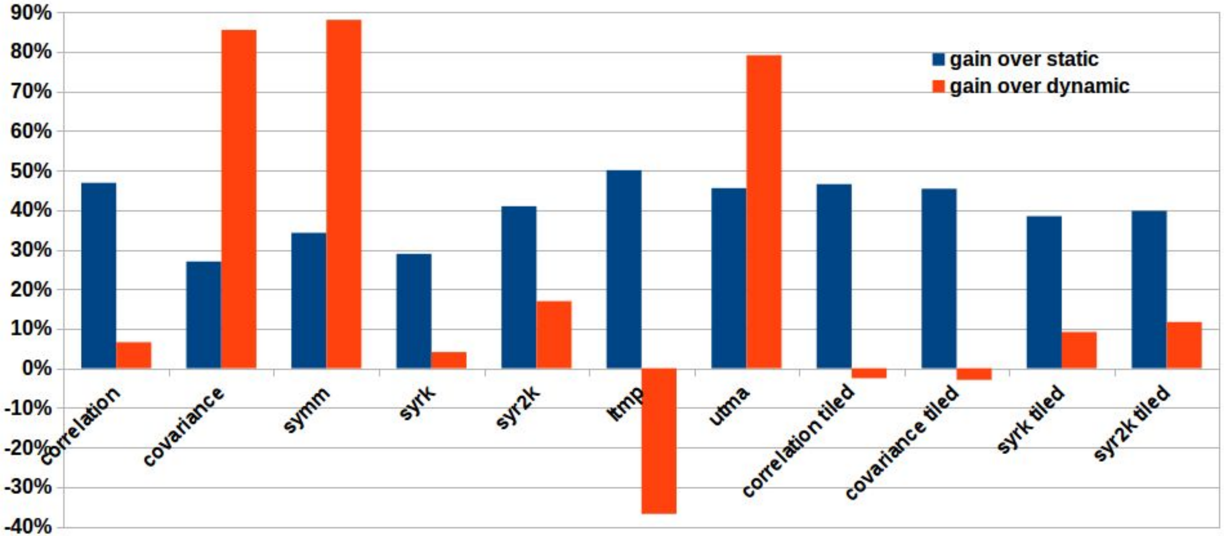
Collapsing of non-rectangular loops

- Using Trahrhe expressions to retrieve the iterator values of the original loops
- + lowering the time overhead of Trahrhe expressions' computations

$$
\begin{array}{l}
first\_iteration = 1; \\
\#\text{pragma omp parallel for private(i,j,k) firstprivate}(first\_iteration) \\
for\ (pc = 1;\ pc \leq N \star (N-1)/2;\ pc++) \\
\quad if\ (first\_iteration)\ \{ \\
\quad\quad i = \text{floor}(\text{-}(\text{sqrt}(4*N*N - 4*N - 8*pc + 9) - 2*N + 1)/2); \\
\quad\quad j = \text{floor}(\text{-}(2*i*N - 2*pc - i*i - 3*i)/2); \\
\quad\quad first\_iteration = 0;\ \} \\
\quad for\ (k = 0;\ k < N;\ k++) \\
\quad\quad\quad a[i][j] += b[k][i] * c[k][j]; \\
\quad\quad a[j][i] = a[i][j]; \\
\quad\quad j++; \\
\quad\quad if\ (j \geq N)\ \{i++;\ j = i + 1;\}\ \}
\end{array}
$$

Collapsing of non-rectangular loops

- 12 threads/cores AMD Opteron 6172, gcc -O3 -fopenmp



- Ph. Clauss, E. Altintas, and M. Kuhn. *Automatic Collapsing of Non-Rectangular Loops*. Parallel and Distributed Processing Symposium (IPDPS), 2017.

Counting-based optimizations

Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

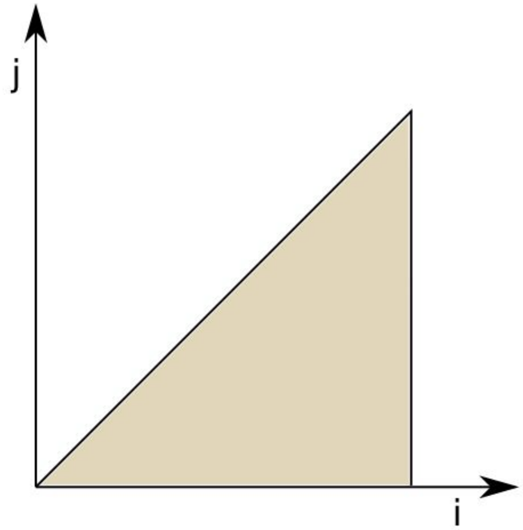
Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Algebraic tiling

- Standard rectangular loop tiling: example `syr2k` (polybench)

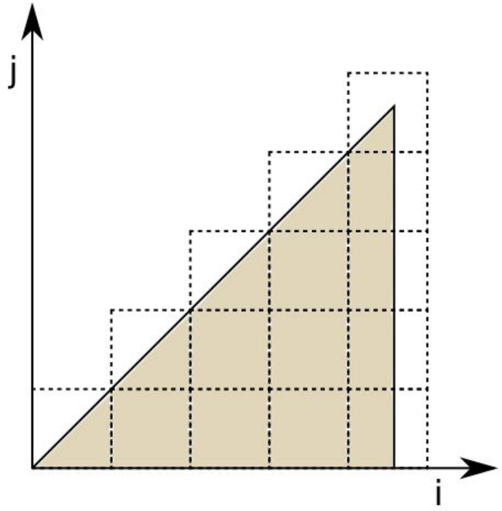
$$
\begin{aligned}
&for\ (i=0;\ i<N;\ i++)\\
&\ for\ (j=0;\ j\leq i;\ j++)\\
&\ \ for\ (k=0;\ k<M; k++)\\
&\ \ \ \ C\,[i]\,[j]+=A\,[j]\,[k]*alpha*B\,[i]\,[k]\\
&\ \ \ \ \ \ \ \ \ \ \ \ \ \ +B\,[j]\,[k]*alpha*A\,[i]\,[k];
\end{aligned}
$$



Algebraic tiling

- Standard rectangular loop tiling: example `syr2k` (polybench)
  - partial tiles issue

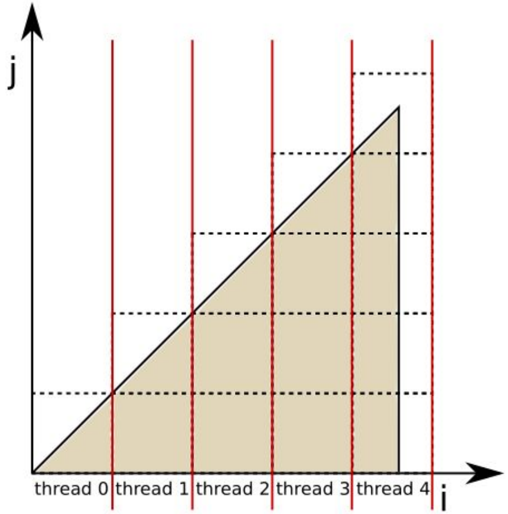
$$
\begin{array}{l}
for\ (it = 0;\ it \le (N-1)/32;\ it++) \\
\ for\ (jt = 0;\ jt \le it;\ jt++) \\
\ \ for\ (kt = 0;\ k \le (M-1)/32; k++) \\
\ \ \ for\ (i = 32 * it;\ i \le \min(N-1, 32 * it + 31);\ i++) \\
\ \ \ \ for\ (j = 32 * jt;\ i \le \min(i, 32 * jt + 31);\ i++) \\
\ \ \ \ \ for\ (k = 32 * kt;\ i \le \min(M-1, 32 * kt + 31);\ i++) \\
\ \ \ \ \ \ C[i][j] += A[j][k] * alpha * B[i][k] \\
\qquad\qquad\qquad + B[j][k] * alpha * A[i][k];
\end{array}
$$



Algebraic tiling

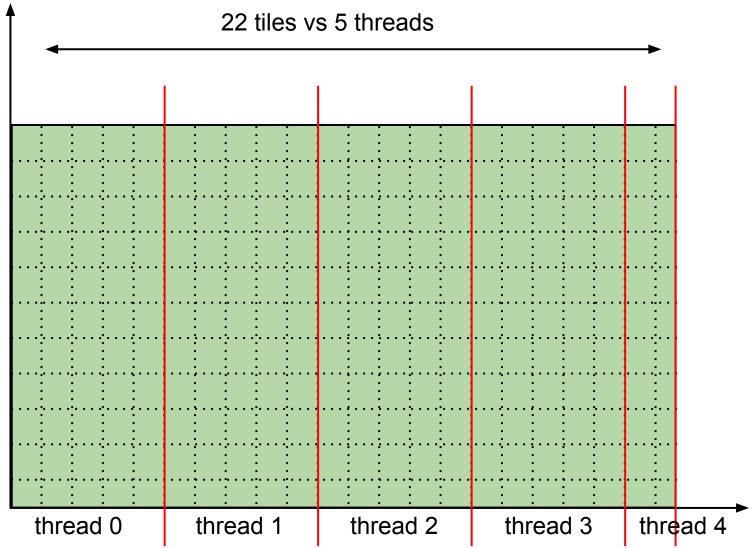
- Standard rectangular loop tiling: example `syr2k` (polybench)
  - load imbalance issue

$$
\begin{array}{l}
\text{\#pragma omp parallel for private(jt,kt,i,j,k)}\\
for\ (it = 0;\ it \leq (N-1)/32;\ it++)\\
\ for\ (jt = 0;\ jt \leq it;\ jt++)\\
\ \ for\ (kt = 0;\ k \leq (M-1)/32; k++)\\
\ \ \ for\ (i = 32*it;\ i \leq \min(N-1, 32*it+31);\ i++)\\
\ \ \ \ for\ (j = 32*jt;\ i \leq \min(i, 32*jt+31);\ i++)\\
\ \ \ \ \ for\ (k = 32*kt;\ i \leq \min(M-1, 32*kt+31);\ i++)\\
\ \ \ \ \ \ C[i][j] += A[j][k]*alpha*B[i][k]\\
\qquad\qquad\qquad +B[j][k]*alpha*A[i][k];
\end{array}
$$



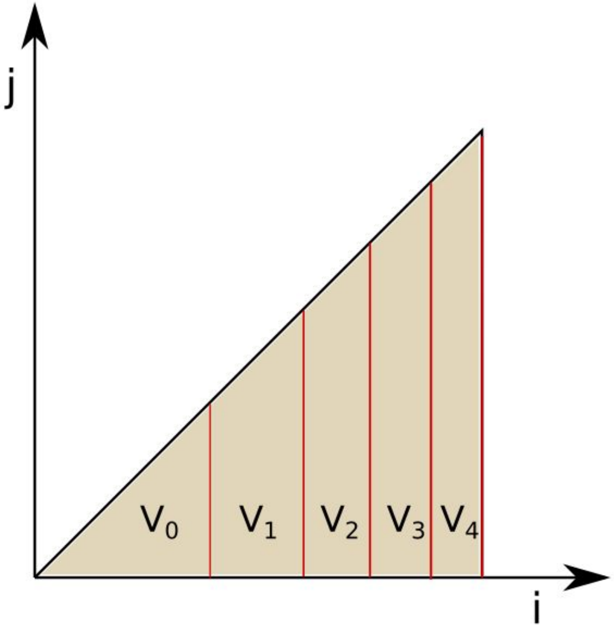
Algebraic tiling

- Standard rectangular loop tiling: example `syr2k` (polybench)
  - load imbalance issue: **even with rectangular domains!**



Algebraic tiling

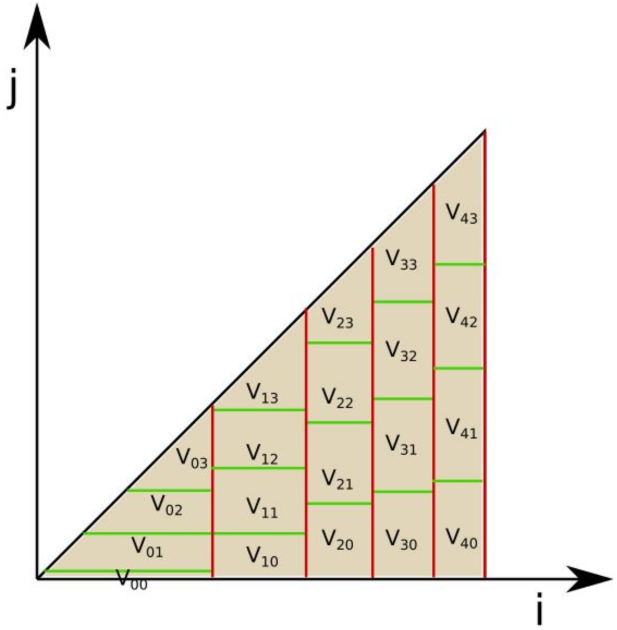
- Slices of quasi-equal volumes (#iterations)



$$V_0 \simeq V_1 \simeq V_2 \simeq V_3 \simeq V_4$$

Algebraic tiling

- Slices of quasi-equal volumes (#iterations)



$$
\begin{aligned}
&V_{00} \simeq V_{01} \simeq V_{02} \simeq V_{03} \\
&\simeq V_{10} \simeq V_{11} \simeq V_{12} \simeq V_{13} \\
&\simeq V_{20} \simeq V_{21} \simeq V_{22} \simeq V_{23} \\
&\simeq V_{30} \simeq V_{31} \simeq V_{32} \simeq V_{33} \\
&\simeq V_{40} \simeq V_{41} \simeq V_{42} \simeq V_{43}
\end{aligned}
$$

Algebraic tiling

- Strategy:
  - **slice the parallel loops in #threads slices of quasi-equal volumes**
    - or a multiple of #threads
  - slice the other loops in any number of slices yielding the best performance
    - or use standard tiling
      - to fix vectorization or overhead issues
- Loop bounds computed on-the-fly
  - target volumes = rank of iterations
  - slice/tile bounds = computed trahrhe expressions

- Clément Rossetti and Philippe Clauss. Algebraic Tiling. IMPACT 2023.

Algebraic tiling

```c
i_pcmax = i_Ehrhart(N, M);
TARGET_VOL_L1 = i_pcmax/DIV1; // DIV1 = #threads

#pragma omp parallel for firstprivate(i_pcmax, TARGET_VOL_L1)  \
            private(i, j, k,  lbi, ubi, lbj, ubj, jt, ubjt, j_pcmax, TARGET_VOL_L2)

for (it = 0; it < DIV1; it++) {
 lbi = trahrhe_i(max(it*(TARGET_VOL_L1+1),1),N, M);
 ubi = trahrhe_i(min((it+1)*(TARGET_VOL_L1+1),i_pcmax),N, M) - 1;
 if (it == DIV1-1) ubi = N-1;

 j_pcmax = j_Ehrhart(N, M, lbi, ubi);
 TARGET_VOL_L2 = j_pcmax/DIV2;

 for (jt = 0; jt < DIV2; jt++) {
  lbj = trahrhe_j(max(jt*(TARGET_VOL_L2+1),1),N, M, lbi, ubi);
  ubj = trahrhe_j(min((jt+1)*(TARGET_VOL_L2+1),j_pcmax),N, M, lbi, ubi) - 1;
  if (jt == DIV2-1) ubj = ubi;

  for (i = max(0,lbi); i < min(N,ubi+1); i++) {
   for (j = max(0,lbj); j <= min(i,ubj); j++) {
    for (k = 0; k <= M-1; k++) {
     C[i][j] += A[j][k]*alpha*B[i][k] + B[j][k]*alpha*A[i][k];
    }
   }
  }
 } /* end for jt */
} /* end for it */
```

Algebraic tiling

- *trahrhe* software: generates the required C header
  - Trahrhe expressions: floating-point computations on complex numbers

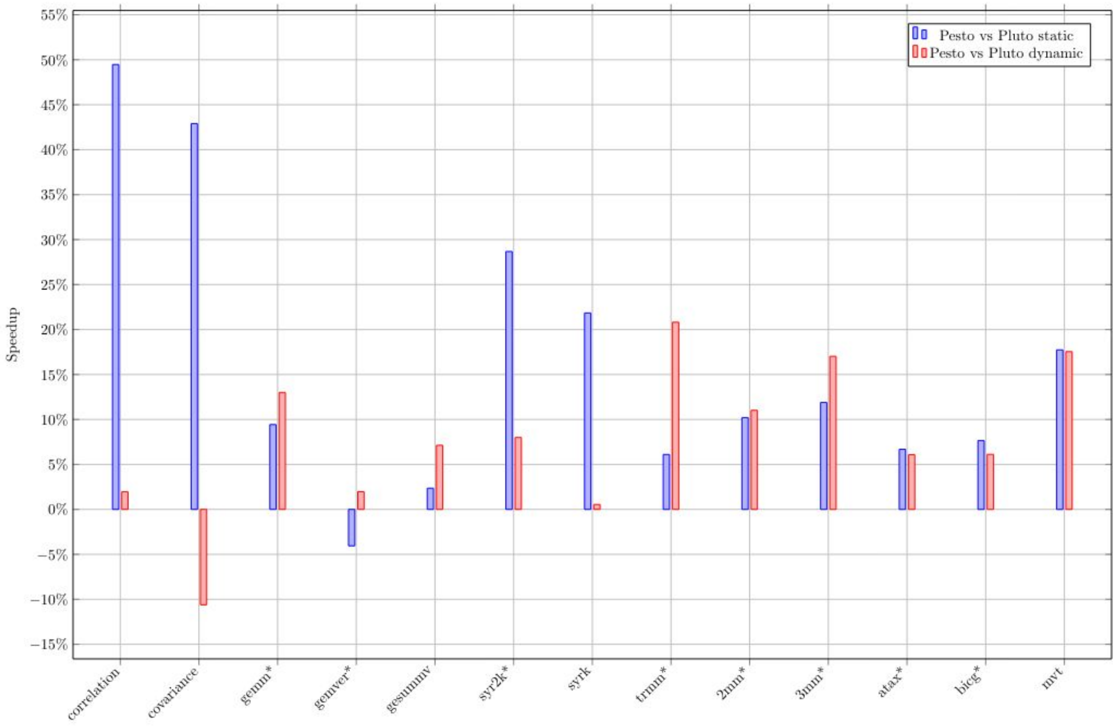
    or

  - Dichotomous search of the roots:
    - often better time performance
    - no precision nor root finding issues
- *pesto* source-2-source algebraic tiler
  - in progress
  - linked to pluto and trahrhe

- https://webpages.gitlabpages.inria.fr/trahrhe

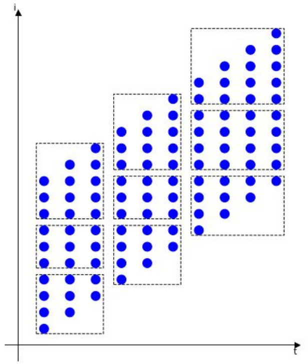
Algebraic tiling

32 threads/cores, best tile sizes/volumes, vectorization activated, dichotomous root finding

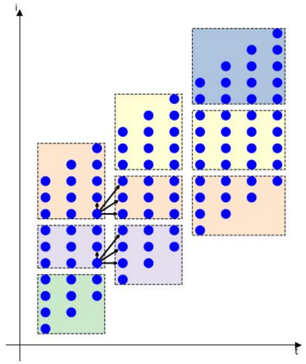


Algebraic tiling

- Standard rectangular tiling
  - some loop kernels (i.e. stencils) require a **tile skewing transformation** to exhibit a parallel loop, which is **impossible with algebraic rect. tiling**



Algebraically tiled domain of seidel-2d



Invalid tile skewing of seidel-2d
Same color means concurrent

- C. Rossetti, A. Hamon, and Ph. Clauss. Algebraic Tiling facing Loop Skewing. IMPACT 2024.

Counting-based optimizations

Loop optimization

- Liveness and memory requirement analyses for array contraction
- Data layout transformation for spatial data locality
- Collapsing of non-rectangular loops
- Algebraic tiling
- Algebraic trapezoidal tiling

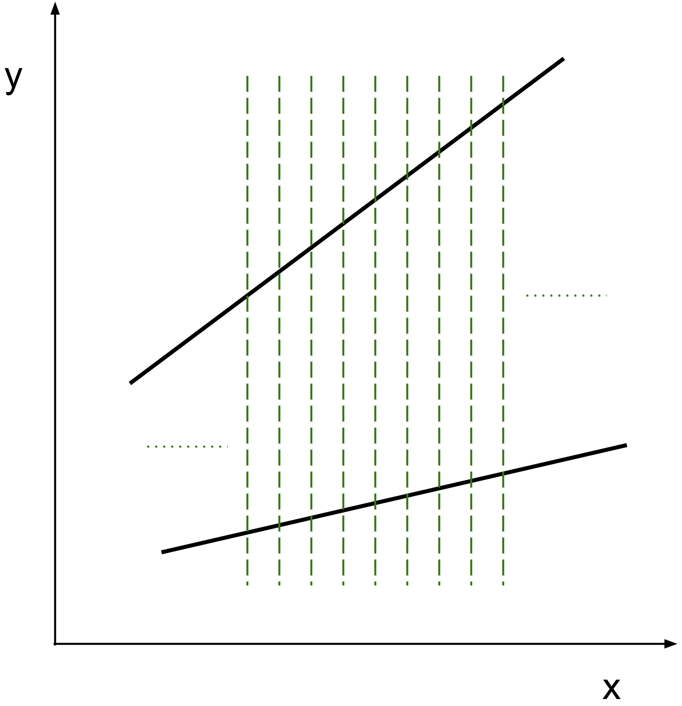
Mathematical support

- Ehrhart polynomials
- Bernstein expansion for polynomial maximization
- Ranking polynomials
- Trahrhe expressions

Algebraic trapezoidal tiling

- For stencil loops
  - as split tiling, diamond tiling, ...
  - but with quasi-equal volumes of the trapezoidal tiles
  - applies even when diamond tiling is not applicable (e.g. seidel-2d)
- Strategy:
  - generate a parallel version of the stencil loop nest through a skewing transformation
    - where the second inner loop is parallel
  - Analyze the dependencies between the parallel fronts
    - the extreme distance vectors define 2 borders of the tiles

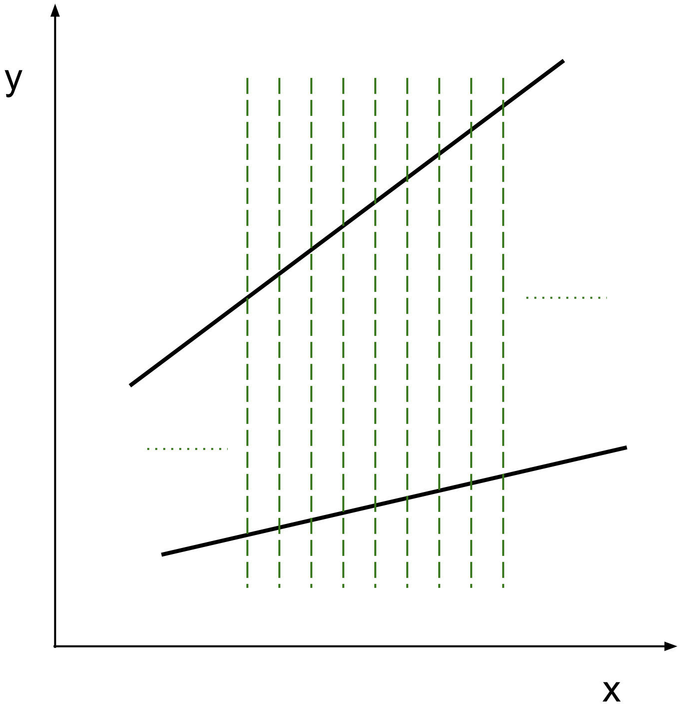
Algebraic trapezoidal tiling



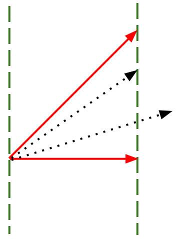
post-skewing parallel fronts

$$
\begin{array}{l}
for\ (x = \ldots;\ x < \ldots;\ x{+}{+}) \\
\ \#\text{pragma omp parallel for ...} \\
\ for\ (y = \ldots;\ y < \ldots;\ y{+}{+}) \\
\quad \{\ldots\}
\end{array}
$$

Algebraic trapezoidal tiling

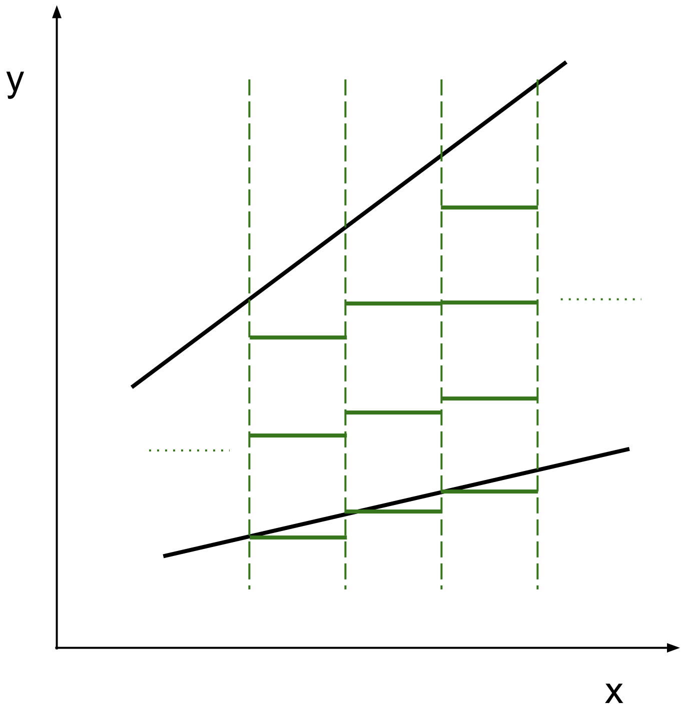


post-skewing parallel fronts



- dependencies between successive fronts and cone of extreme distance vectors
- constraint: one extreme vector must be (1,0)

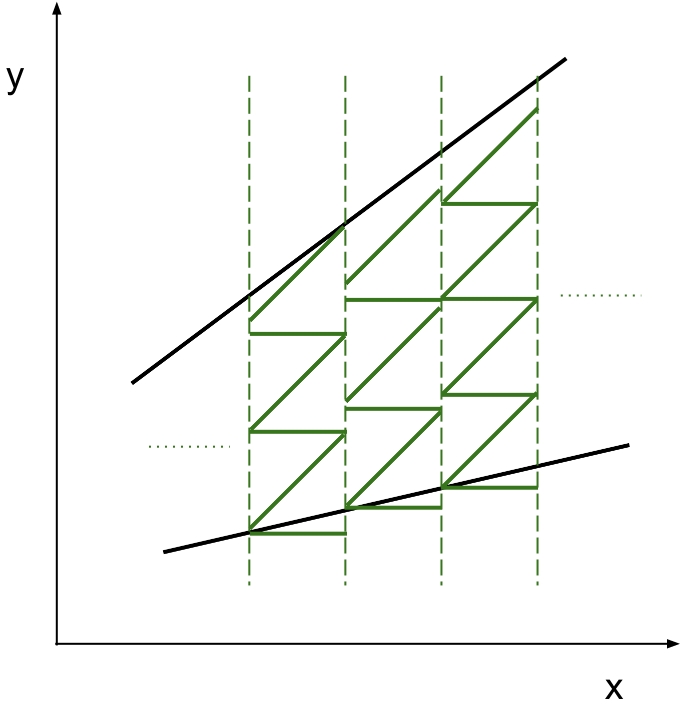
Algebraic trapezoidal tiling



algebraic rectangular tiling inside x-slices of constant width with a fixed target volume per tile

constraint:
tiles height ≥ slope of the diagonal extreme dependence vector ✕ slice width

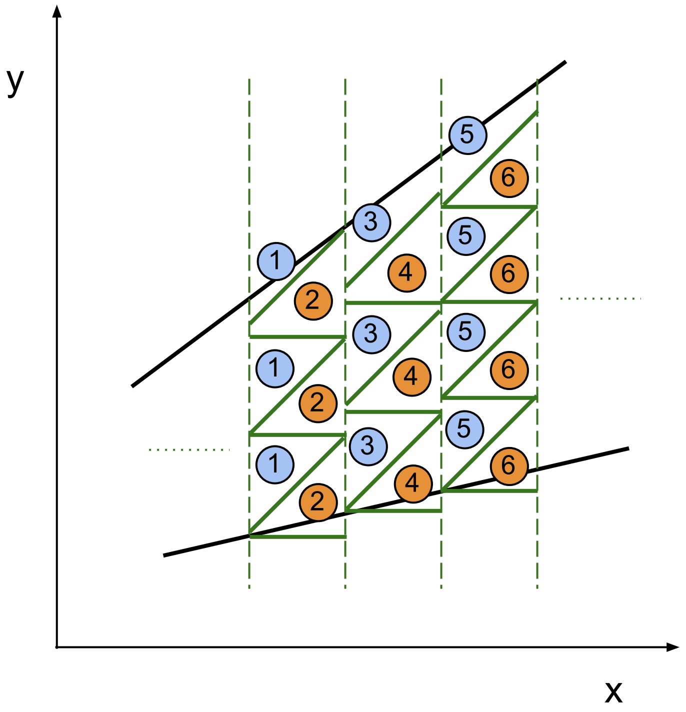
Algebraic trapezoidal tiling



split of the rectangular tiles along the direction of the diagonal extreme dependence vector

⇒ parallel trapezoidal tiles

Algebraic trapezoidal tiling



parallel fronts of independent trapezoidal tiles of quasi equal volumes

Algebraic trapezoidal tiling: seidel-2d

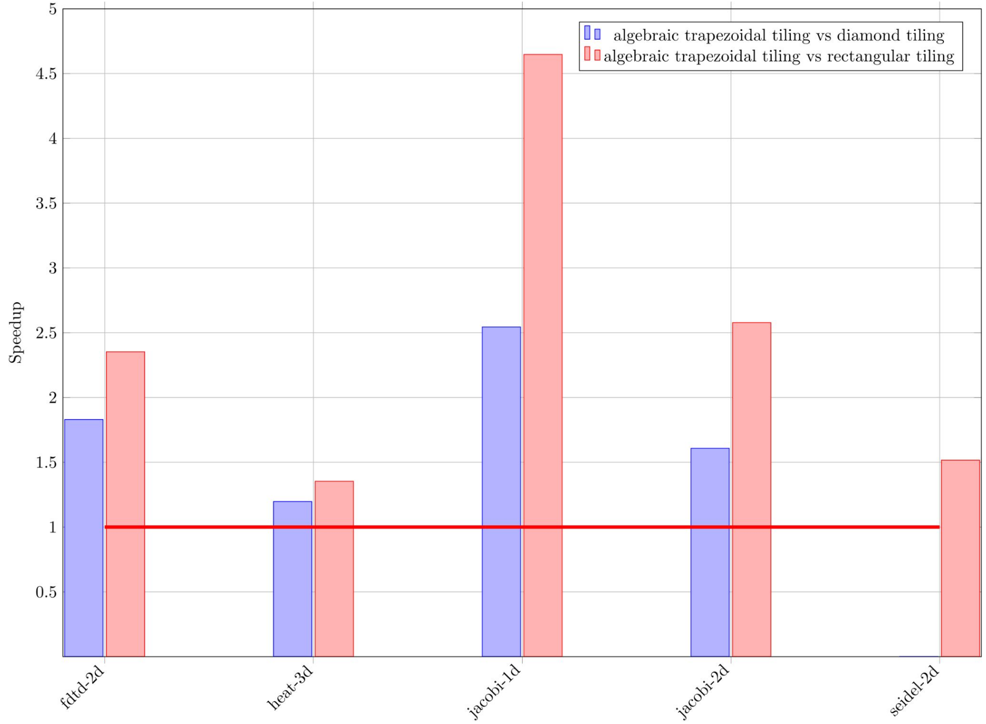
```c
#define SliceWidthx 6
#define NTHREADS 64
#define TARGET_VOL 143560
#define SliceWidthz 2

/* SliceCountx: #outermost slices */
int SliceCountx=ceild(((2*_PB_TSTEPS+_PB_N-4-1)+1),SliceWidthx);
for (xt = 0; xt < SliceCountx; xt++)  {
  /* lbx, ubx: bounds of outermost slices */
  lbx=1+xt*SliceWidthx;
  ubx=min(1+(xt+1)*SliceWidthx-1,2*_PB_TSTEPS+_PB_N-4);
  /* y_pcmax: volume of the current outermost slice */
  y_pcmax = y_Ehrhart(lbx,ubx);
  /* DIV: #rectangular tiles in the current slice
          whose volume >= TARGET_VOL and DIV<=NTHREADS */
  DIV=min(max(floord(y_pcmax,TARGET_VOL),1),NTHREADS);
  /* TILE_VOL: actual target volume for the rectangular tiles */
  TILE_VOL=y_pcmax/DIV;
  /* FRONT: alternating parallel fronts of trapezoidal tiles */
  for (FRONT=0; FRONT<=1; FRONT++) {
#pragma omp parallel for private(lby,uby,x,LBOUND,UBOUND,y,z,OFFSET,t,i,j,zt,lbz,ubz) \
                         firstprivate(TILE_VOL,DIV,FRONT,y_pcmax)
    /* enumerate the rect. tiles of quasi-equal volumes */
    for (yt=0; yt<DIV; yt++) {
      lby=y_trahrhe_y(max(yt*TILE_VOL,1), lbx, ubx);
      uby=y_trahrhe_y(min((yt+1)*TILE_VOL,y_pcmax), lbx, ubx)-1;
      if (yt==DIV-1) uby=min(floord(ubx+_PB_N-2,2),ubx);
      /* check if tile height >= tile width */
      if ((uby-lby+1 < SliceWidthx) && (yt != DIV-1) && (DIV>1)) {
        fprintf(stderr, "Target volume too small!\n");
        exit(1);
      }
      /* compute where the rect. tile will be cut */
      OFFSET= max(floord((uby-lby+1)-SliceWidthx,2),0);

      int SliceCountz=ceild((ubx+_PB_N-2) - (lbx+1) +1,SliceWidthz);
```

Algebraic trapezoidal tiling: seidel-2d

```
      for (zt=0; zt<SliceCountz; zt++) {
        lbz=lbx + 1 + zt*SliceWidthz;
        ubz=min(lbx + 1 + (zt+1)*SliceWidthz-1,ubx+_PB_N-2);
        /* inner tile loops */
        for (x = max(1,lbx); x <= min(2*_PB_TSTEPS+_PB_N-4,ubx); x++)  {
          int lbp=max(ceild(x+1,2),x-_PB_TSTEPS+1);
          int ubp=min(floord(x+_PB_N-2,2),x);
          /* Compute the bounds of the trapezoidal tiles to cut the rect. tiles */
          if (FRONT==0) {
           LBOUND=max(max(lby,lbp),max(ceild(max(1,lbx)+1,2),max(1,lbx)-_PB_TSTEPS+1))+(x-max(1,lbx))+OFFSET);
           UBOUND=min(ubp,uby);
          }
          else {
            LBOUND=max(lbp,lby);
            UBOUND=min(min(ubp,uby),max(max(lby,lbp),max(ceild(max(1,lbx)+1,2),max(1,lbx)-_PB_TSTEPS+1))+(x-max(1,lbx))+OFFSET-1);
          }
          /* enumerate the current trapezoidal tile */
          for (y=LBOUND; y <= UBOUND; y++)
            for (z = max(x + 1,lbz); z <= min(x + 3998,ubz); z += 1) {
              A[(-x + 2 * y)][(-x + z)] = (A[(-x + 2 * y)-1][(-x + z)-1] + A[(-x + 2 * y)-1][(-x + z)] + A[(-x + 2 * y)-1][(-x + z)+1]
                     + A[(-x + 2 * y)][(-x + z)-1] + A[(-x + 2 * y)][(-x + z)] + A[(-x + 2 * y)][(-x + z)+1]
                     + A[(-x + 2 * y)+1][(-x + z)-1] + A[(-x + 2 * y)+1][(-x + z)] + A[(-x + 2 * y)+1][(-x + z)+1])
                     /SCALAR_VAL(9.0);
            }
        }
      }
    }
  }
}
```

Algebraic trapezoidal tiling vs Diamond tiling

32 threads/cores, best tile sizes/volumes, vectorization activated, dichotomous root finding



seidel-2d: diamond tiling not applicable (standard skewing)

Conclusion

- Counting-based optimizations
  - Interesting speedups despite some time overhead
    - runtime tile bounds, floating-point computations, dichotomous root finding, …
    - may still be improved
  - Interesting perspectives: algebraic schedule

- **Runtime** counting-based optimizations
  - for “polyhedral-behaving” kernels
  - made possible thanks to Alain Ketterlin’s work

Thank you!

Philippe Clauss
University of Strasbourg + Inria
France