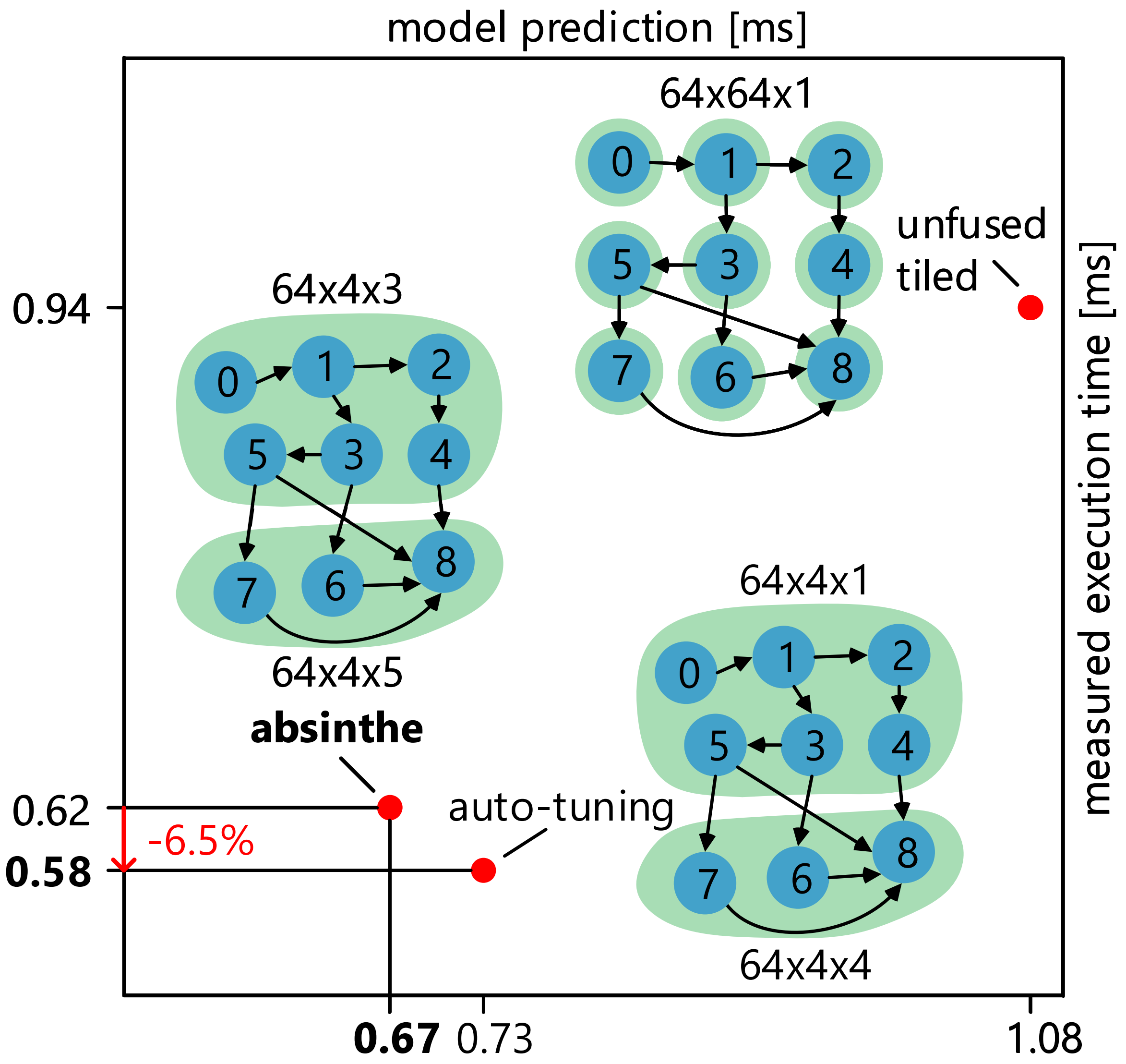
Tobias Gysi, Tobias Grosser, and Torsten Hoefler

Absinthe: Learning an Analytical Performance Model to Fuse and Tile Stencil Codes in One Shot

COSMO Atmospheric Model

- Regional atmospheric model used by 7 national weather services
- Implements many different stencil programs

Optimizing the Fastwaves Kernel from the COSMO Atmospheric Model



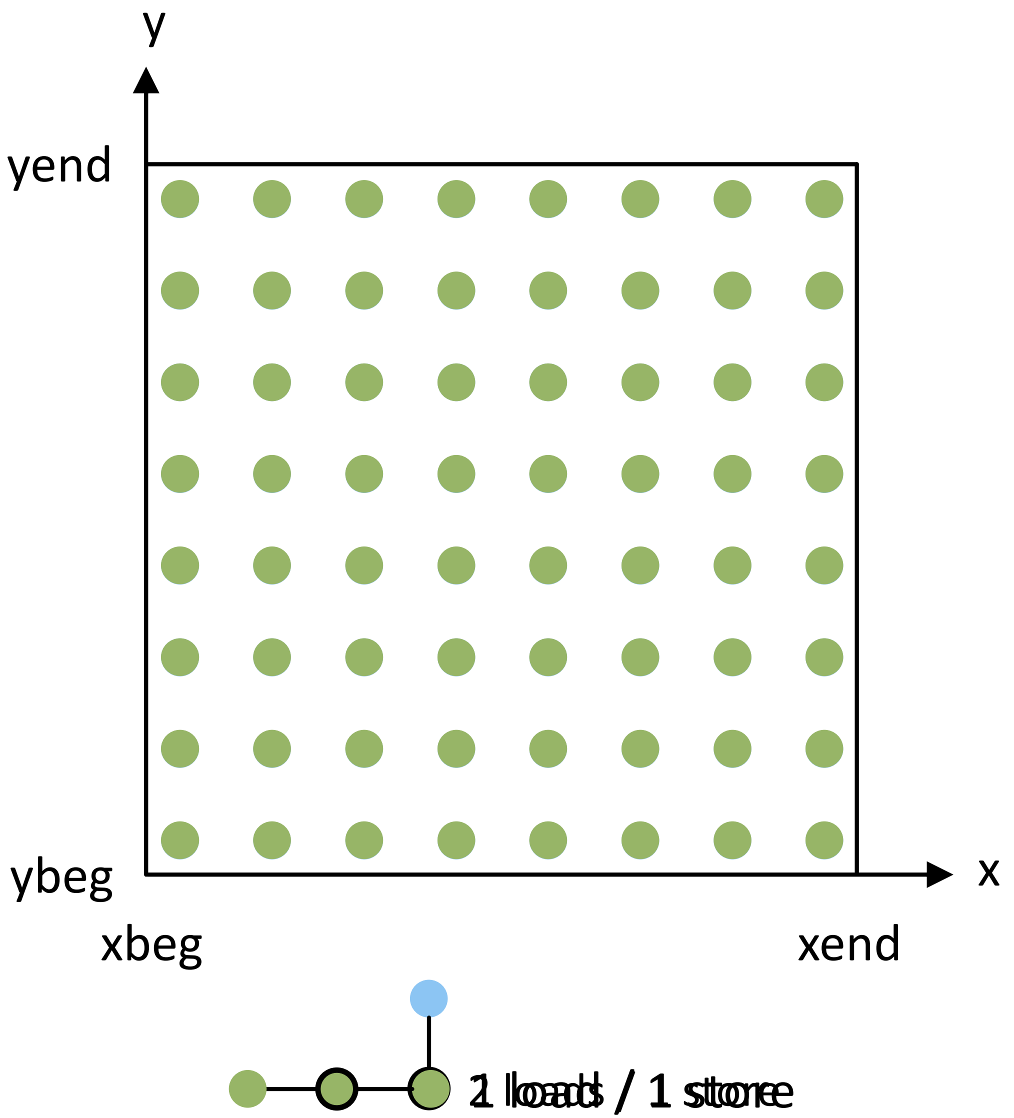
Michael Baldauf, Axel Seifert, Jochen Förstner, Detlev Majewski, Matthias Raschendorfer, and Thorsten Reinhardt, *Operational Convective-Scale Numerical Weather Prediction with the COSMO Model: Description and Sensitivities*. 2011.

Stencil Programs Execute Multiple Stencils in Sequence

```
for (int y = ybeg; y < yend; y++)
  for (int x = xbeg; x < xend; x++)
    A(x,y) = I(x,y) + I(x-1,y) + I(x+1,y);

for (int y = ybeg; y < yend; y++)
  for (int x = xbeg; x < xend; x++)
    B(x,y) = A(x,y+1) + A(x,y);
```

- element-wise computation
- position independent access pattern

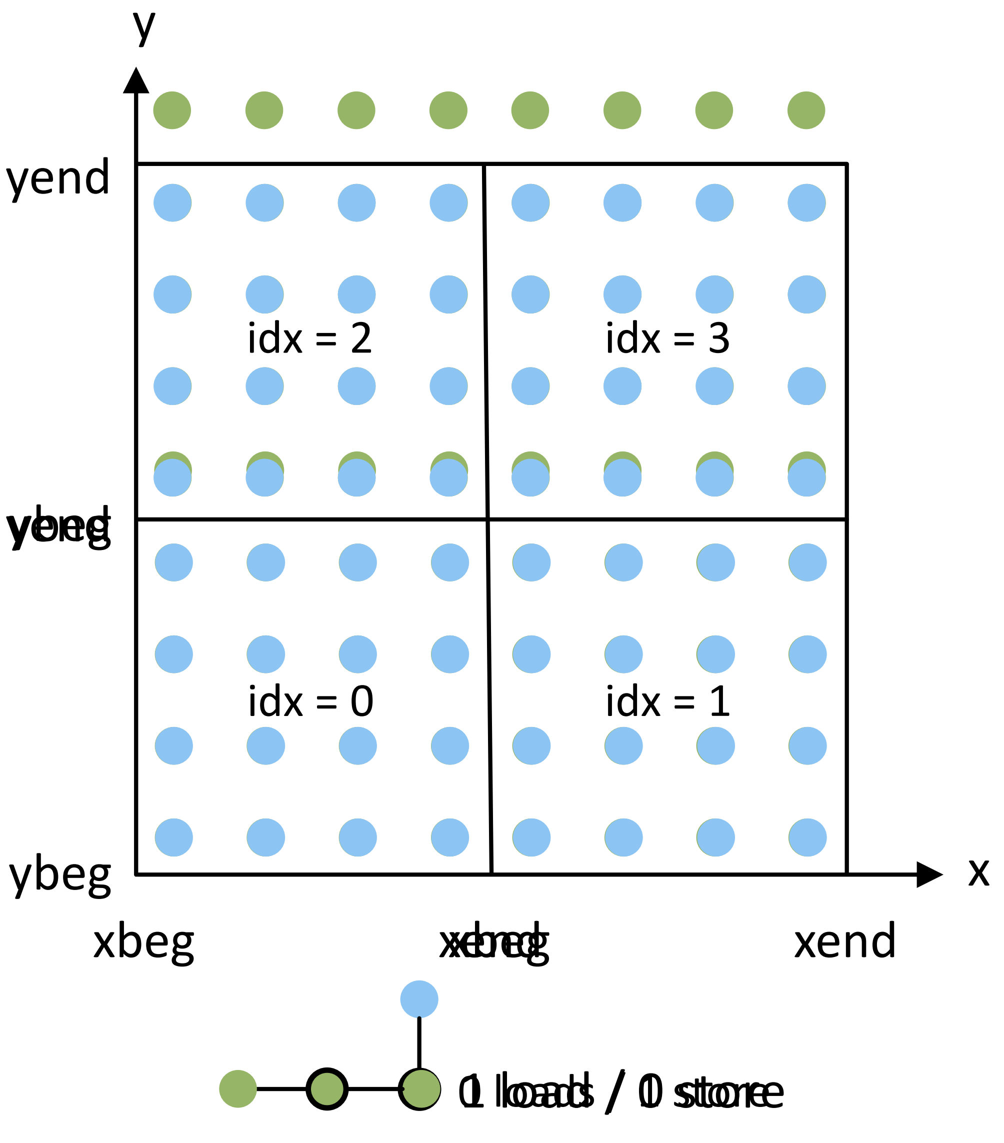


Loop Tiling and Loop Fusion

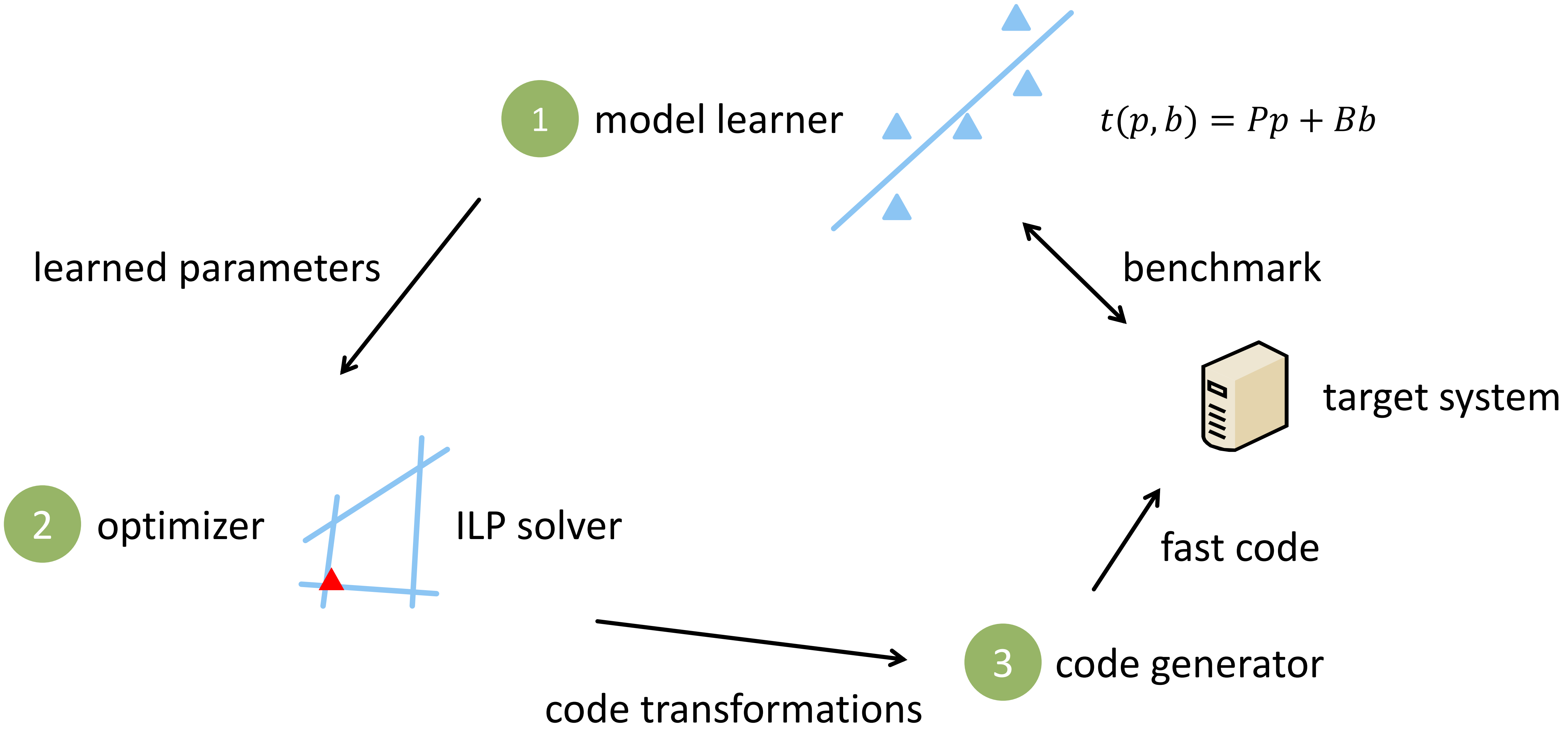
```
for (int idx = 0; idx < 4; ++idx) {
  int xbeg = tiles[idx].xbeg;
  int xend = tiles[idx].xend;
  int ybeg = tiles[idx].ybeg;
  int yend = tiles[idx].yend;
  Buffer A(xbeg, xend, ybeg, yend+1);

  for (int y = ybeg; y < yend+1; ++y)
    for (int x = xbeg; x < xend; ++x)
      A(x,y) = I(x,y) + I(x-1,y) + I(x+1,y);

  for (int y = ybeg; y < yend; y++)
    for (int x = xbeg; x < xend; x++)
      B(x,y) = A(x,y+1) + A(x,y);
}
```

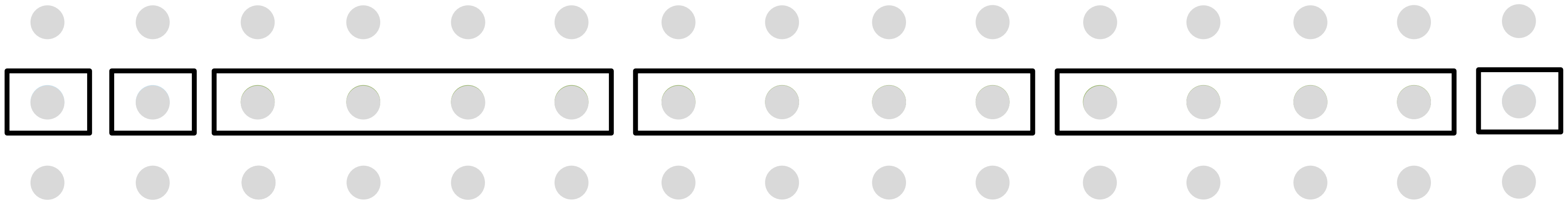


Architecture Overview



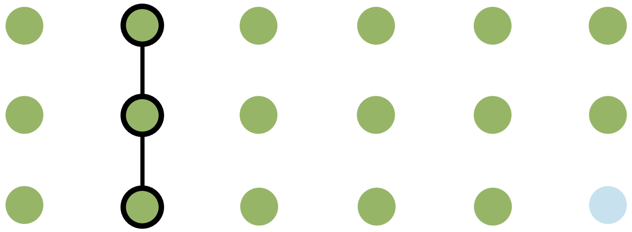
Performance Model Ideas

- execution time of innermost loop



scalar peel loops · vectorized loop body

- memory accesses dominate the execution time

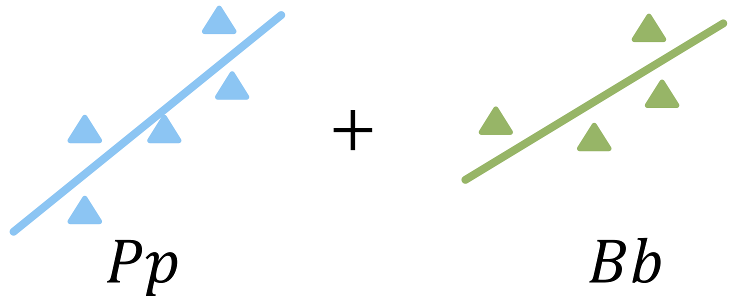


fast memory (L1 cache) · slow memory (L3 cache/DDR)

Performance Model Design

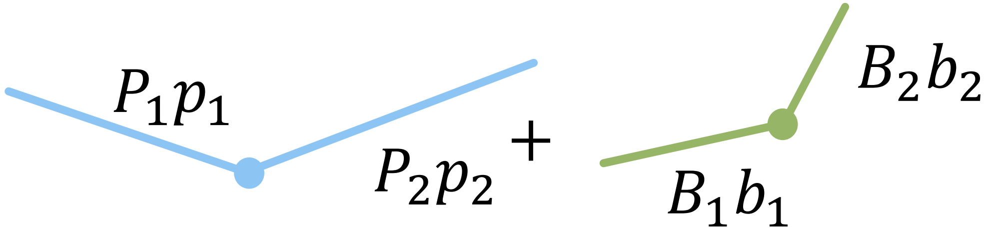
- linear cost functions for peel and body cost

$$t = Pp + Bb$$



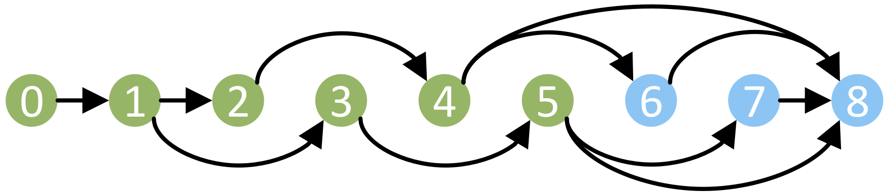
- slow and fast memory

$$t = \max(P_1p_1, P_2p_2) + \max(B_1b_1, B_2b_2)$$



- model the entire program

$$t = \sum_{i=0..8} t_i$$



Evaluating the Fast Memory Model

- \# cache accesses

$$p^f = (n^x + 1)(D^y + e^y n^y)$$

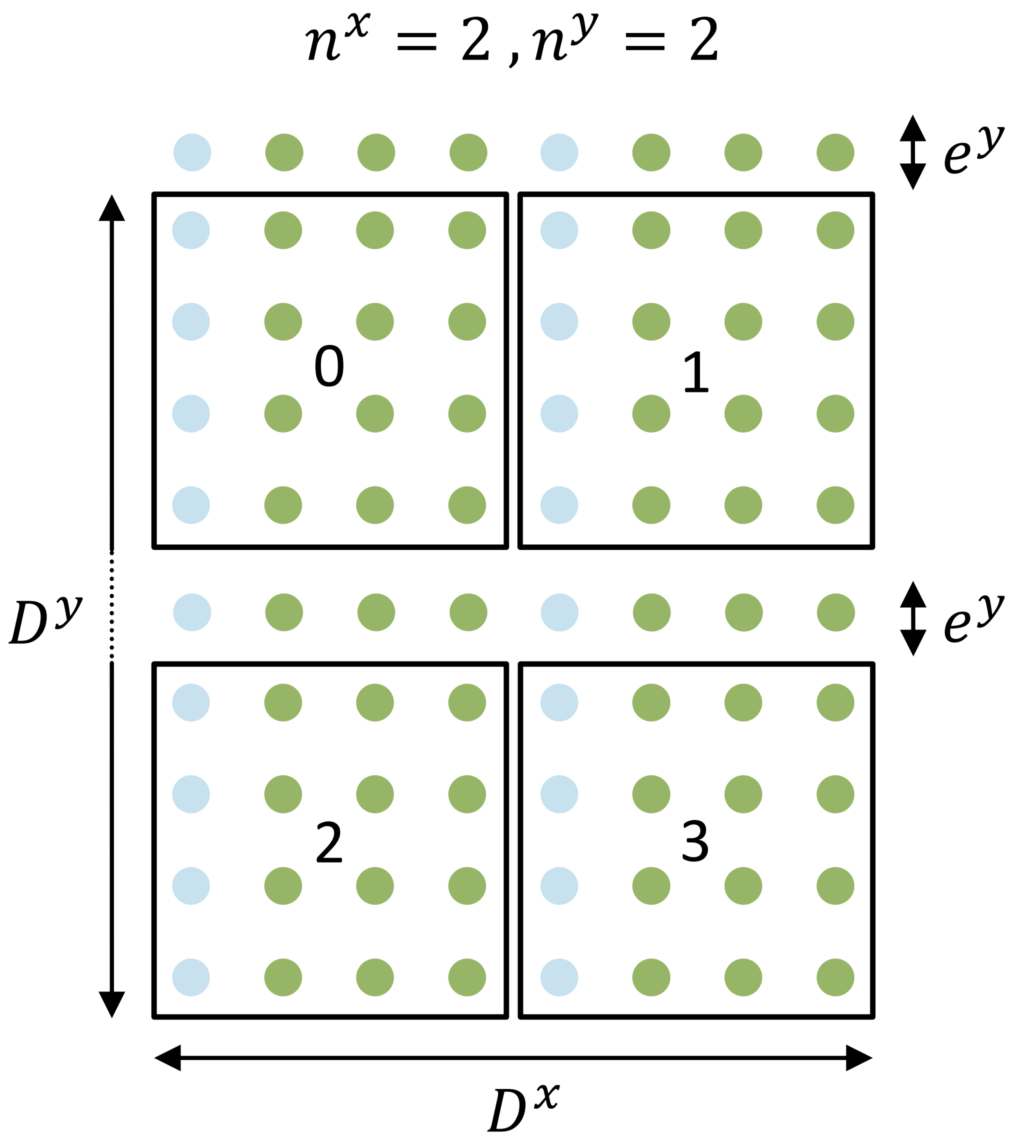
$$b^f = D^x D^y + D^x e^y n^y$$

3 loads / 1 store

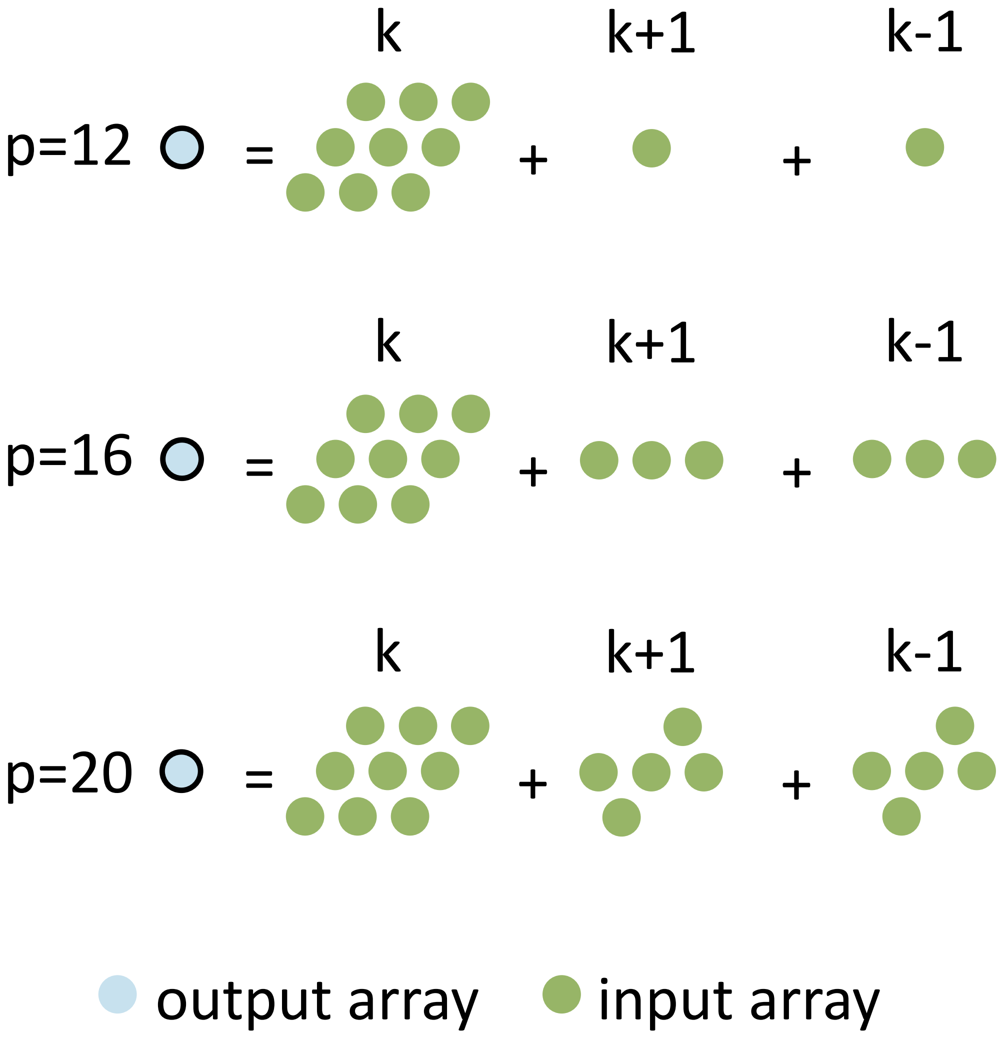
- estimated execution time

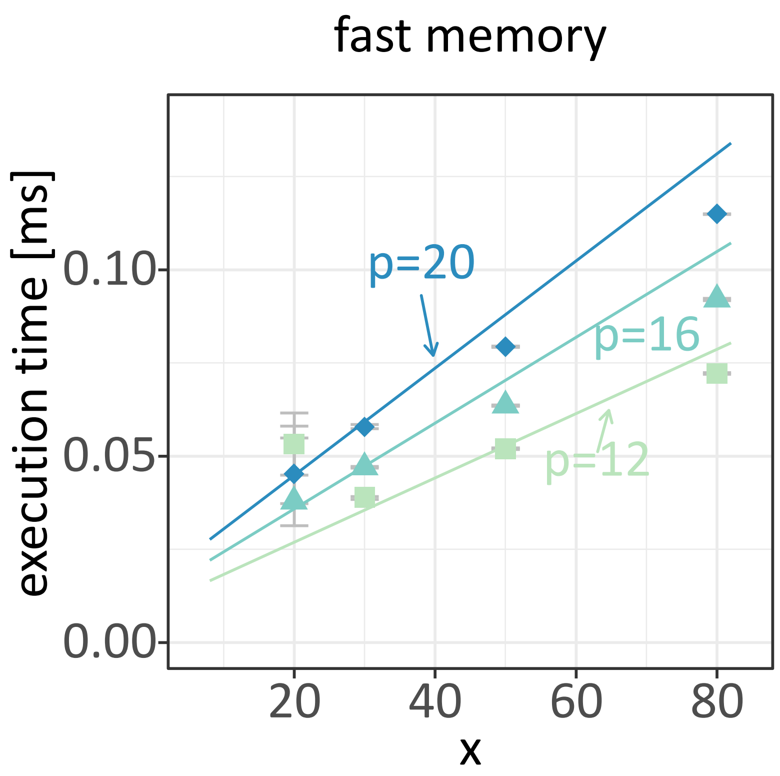
$$t = P^f p^f + B^f b^f$$

learn the model parameters P^f, B^f



Learning the Fast Memory Model



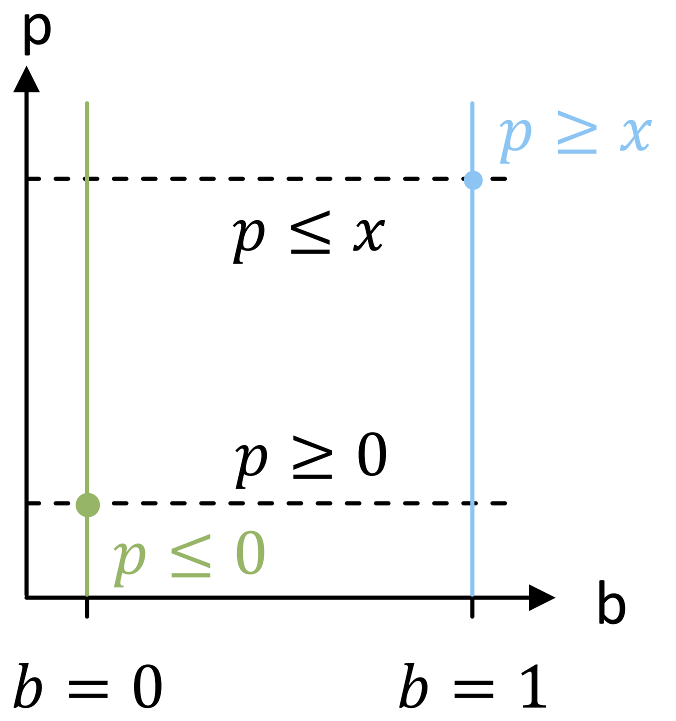


$$\left(P^f, B^f\right) = \underset{(P,B)\in\mathbb{R}}{\operatorname{argmin}} \sum_{r\in[0,R]} \left|(Pp_r - Bb_r) - t_r\right|$$

Linear Multiplication of Bounded Integer Variables

- the binary product $p = xb$ given the upper bound X

result	0	x
limit range	$\mathbf{0 \leq p \leq x}$	
force result	$\boldsymbol{p - Xb \leq 0}$	$\boldsymbol{p - x - Xb \geq -X}$



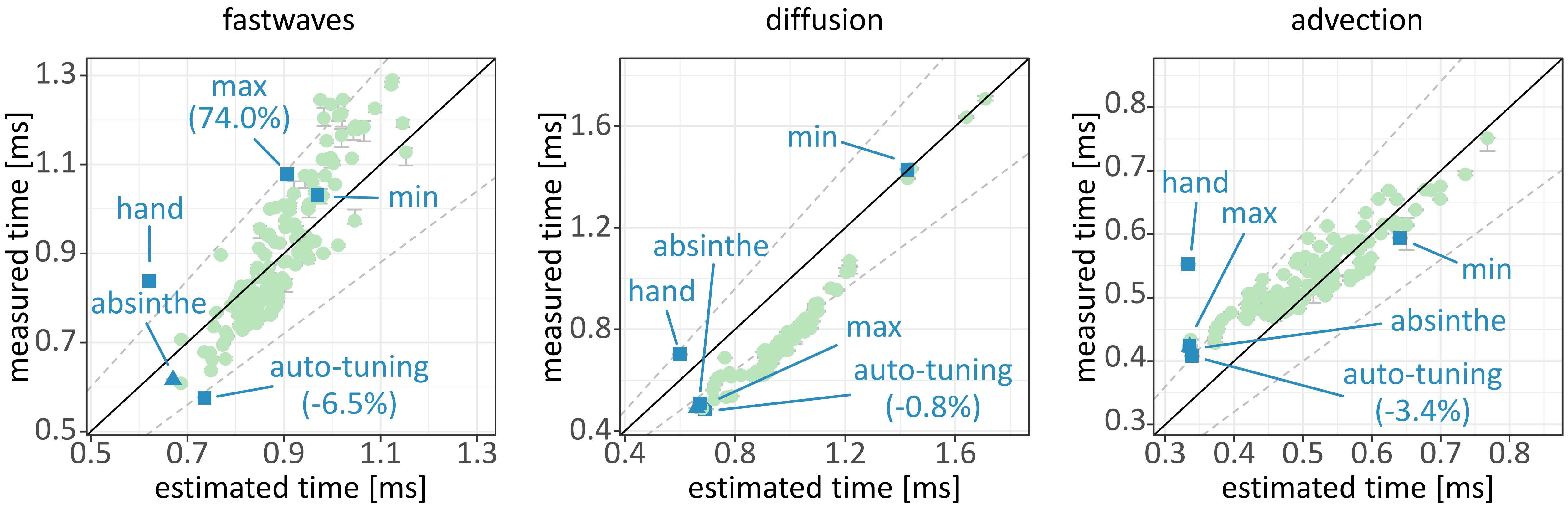
- the integer product $p = xy$ given the upper bounds X and Y

binary representation
$$y = \sum_{i=0}^{\lfloor \log_2(Y) \rfloor} 2^i y_i$$

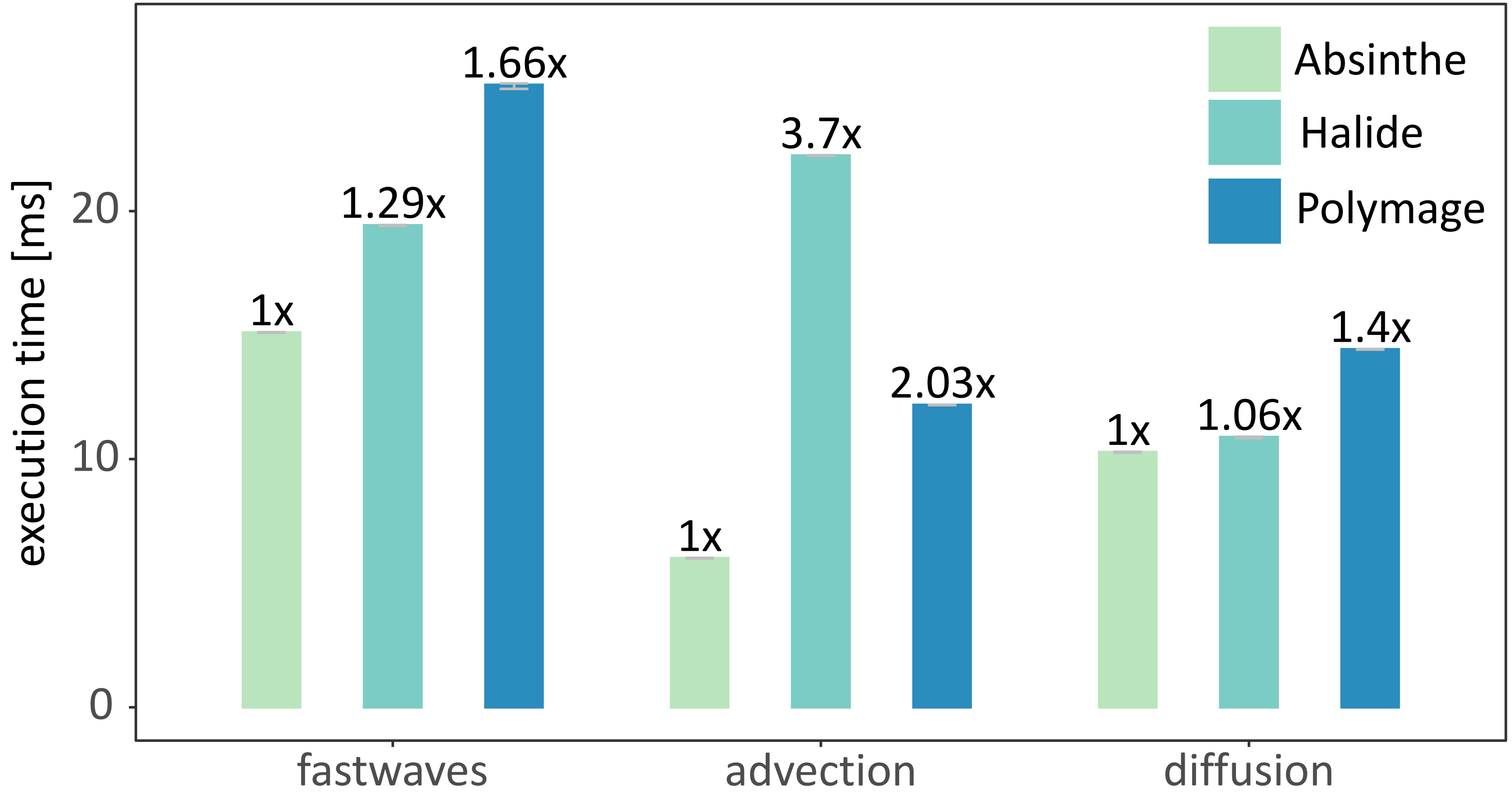
sum binary products
$$p = \sum_{i=0}^{\lfloor \log_2(Y) \rfloor} 2^i x y_i$$

https://blog.adamfurmanek.pl/2015/09/26/ilp-part-6/

Comparison to Auto-tuning, Heuristics, Hand-tuned, and Random Variants



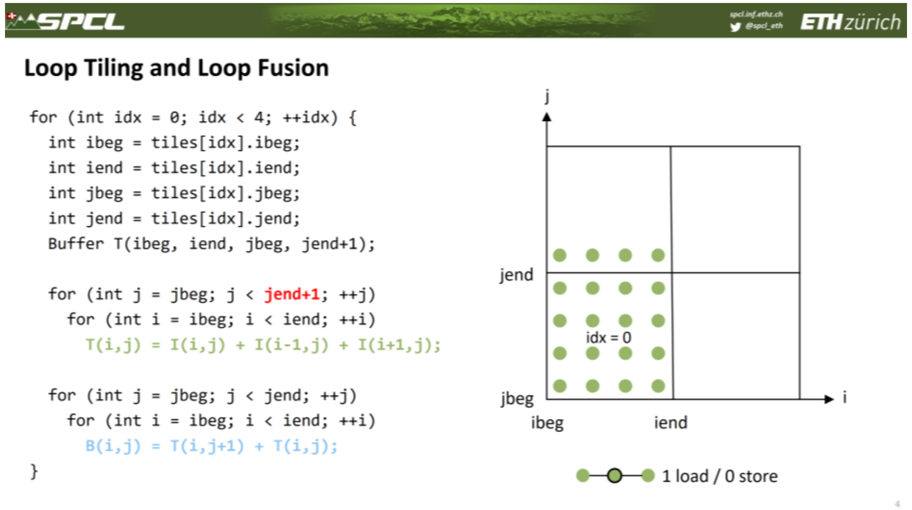
Comparison to Halide and Polymage



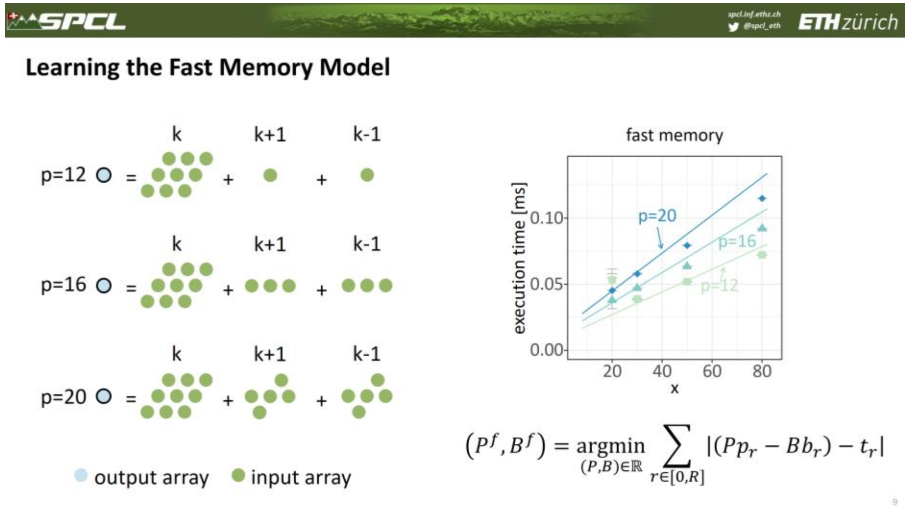
R. T. Mullapudi, A. Adams, D. Sharlet, J. Ragan-Kelley, and K. Fatahalian, *Automatically scheduling halide image processing pipelines*. 2016.
A. Jangda and U. Bondhugula, *An effective fusion and tile size model for optimizing image processing pipelines*. 2018.

Conclusions

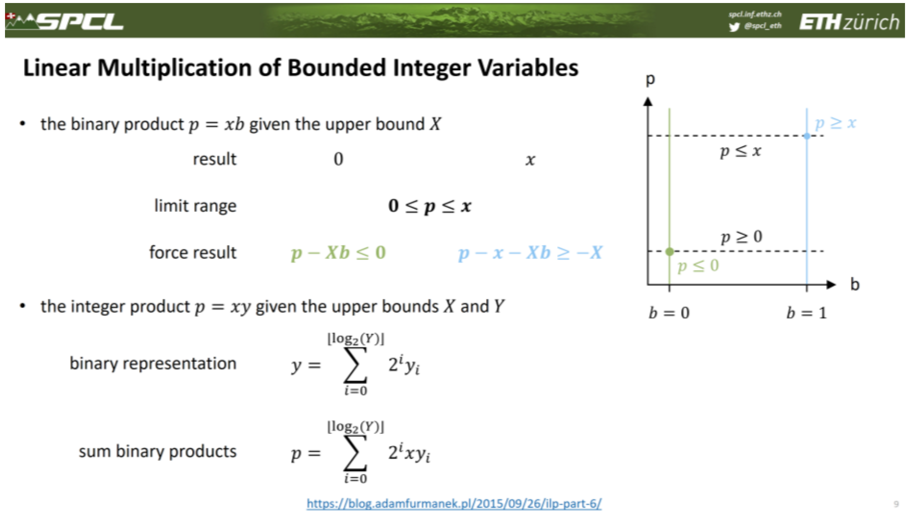
loop fusion and loop tiling



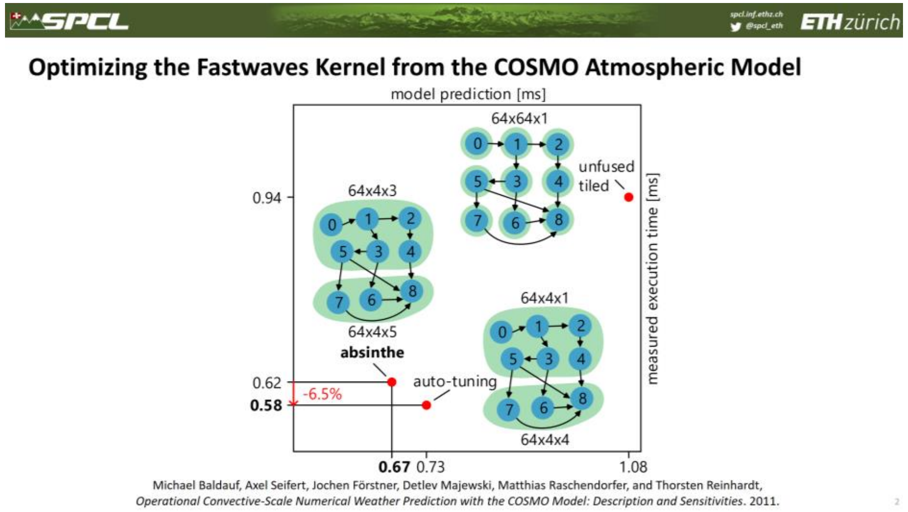
learned performance model



integer linear programming

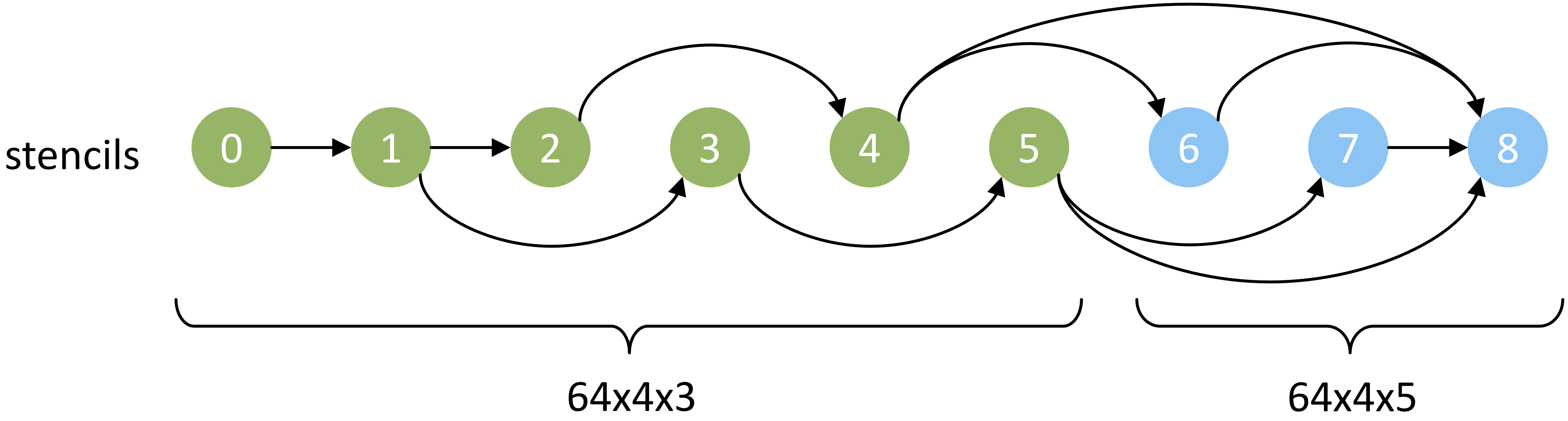


close to auto-tuning



Backup Slides

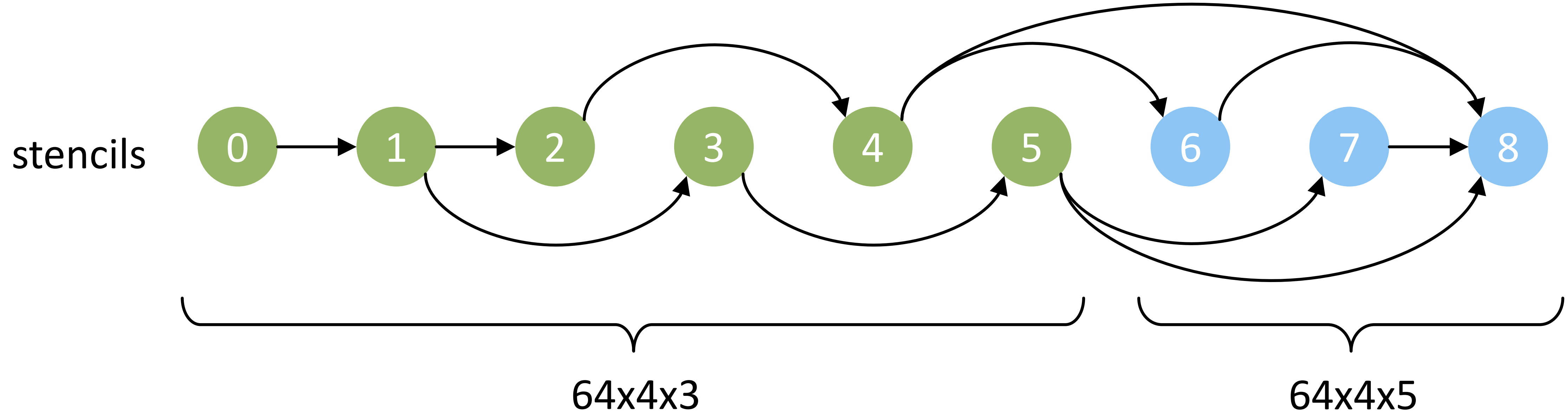
Model the Space of Possible Code Transformations



fusion choices $g_0 = 0$ $g_1 = 0$ $g_2 = 0$ $g_3 = 0$ $g_4 = 0$ $g_5 = 0$ $g_6 = 1$ $g_7 = 1$ $g_8 = 1$

$$0 \leq g_{i+1} - g_i \leq 1 \quad \forall i \in [0,7]$$

Model the Space of Possible Code Transformations



tile sizes

$n_0^x = 1$, $n_0^y = 16$, $n_0^z = 20$ … $n_5^x = 1$, $n_5^y = 16$, $n_5^z = 20$

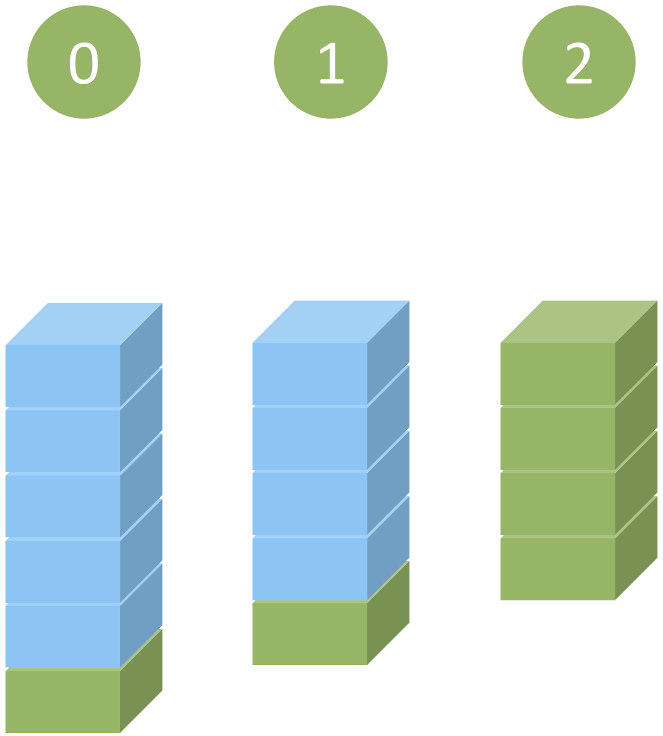
$n_6^x = 1$, $n_6^y = 16$, $n_6^z = 12$ … $n_8^x = 1$, $n_8^y = 16$, $n_8^z = 12$

equality constraints

$$1 \le n_i^x \le D^x,\ 1 \le n_i^y \le D^y,\ 1 \le n_i^z \le D^z \quad \forall i \in [0,8]$$

Limit the Cache Utilization

stencils



$F_{02} = 6 \quad F_{12} = 5 \quad F_{22} = 4$

$$f_2 \geq F_{22}$$

$$f_2 + F_{12}(g_2 - g_1) \geq F_{12}$$

$$f_2 + F_{02}(g_2 - g_0) \geq F_{02}$$

$$C n_2^x n_2^y n_2^z - f_2 \geq 0$$