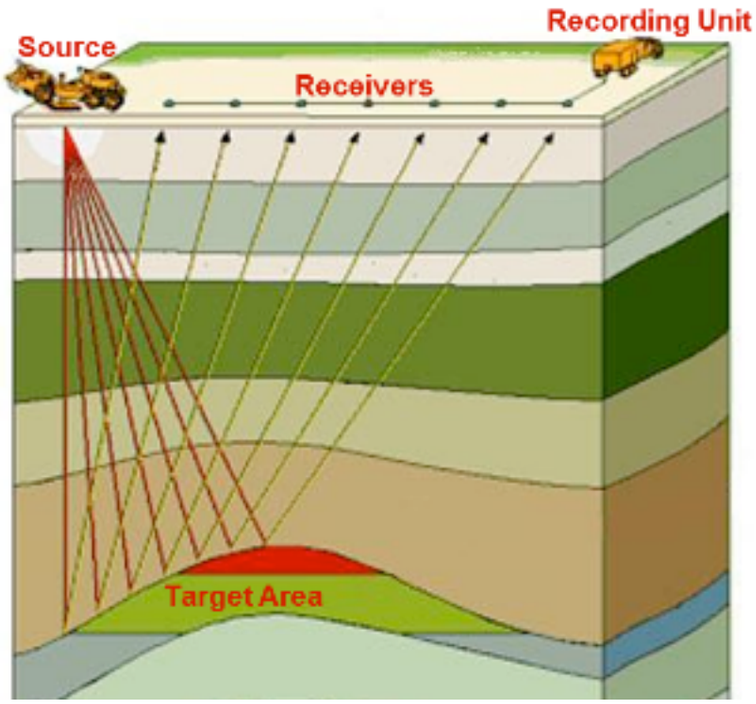
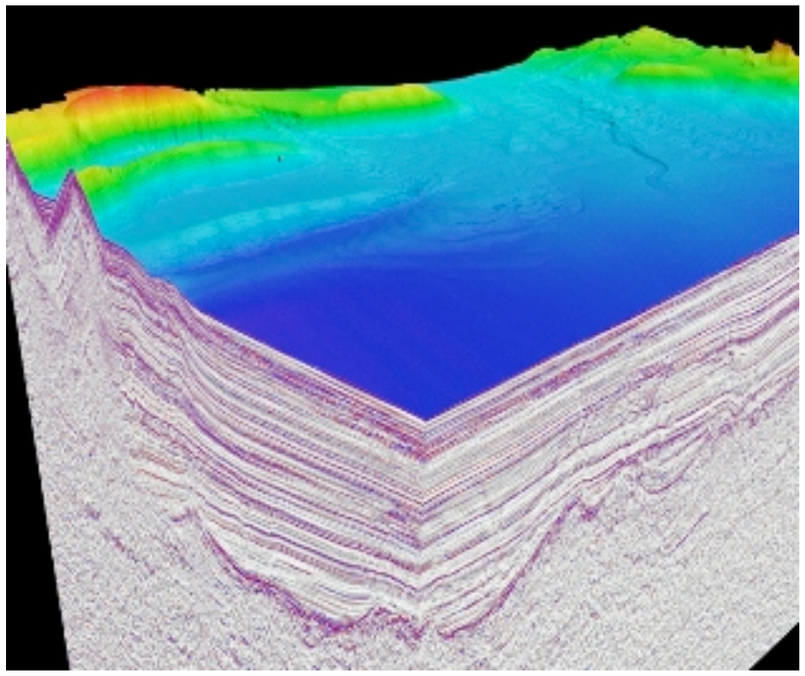
Accelerate The Stencil Computation by Time-skewing

Muhong Zhou

*The importance of forward modeling:

1. It can be used to generate synthetic seismograms with the acquired seismic data
2. To develop models addressing the problems of structure and stratigraphy during the interpretation of seismic data
3. To simulate the seismic response that would occur if the target objectives were met
4. ...





* http://www.arcis.com/Satinder/PDFs/INTRODUCTION%20TO%20TLE%20SPECIAL%20SECTIONS/Intro_Seismic_Modeling_TLE_May09.pdf
[figures]: http://www.tgs.com/geophysical/onshore-seismic-data/seismic-acquisition-basics.aspx

The wave propagation problem (forward modeling)

Wave Equation:

$$\frac{1}{C^2}U_{tt}(t,\vec{X}) - \bigtriangledown^2 U(t,\vec{X}) = 0 \quad t \in [0,T], \vec{x} \in [0,X] \times [0,Y] \times [0,Z]$$

Initial Condition

$$\frac{\partial U}{\partial t}(0,\vec{X}) = 0, U(0,\vec{X}) = \begin{cases} 1 & \vec{X} = (\frac{X}{2}, \frac{Y}{2}, \frac{Z}{2}) \\ 0 & else \end{cases}$$

Boundary Condition

$$U(0,\vec{X}) = 0 \; if \; x = 0 \; or \; X, y = 0 \; or \; Y, z = 0 \; or \; Z$$

Finite difference scheme(2-4) and corresponding stencil codes for the elastic wave equation

$$
\begin{aligned}
&\tfrac{1}{C^2}\tfrac{U(t+\Delta t,\vec{X})-2U(t,\vec{X})+U(t-\Delta t,\vec{X})}{\Delta t^2} \\
= \;& (\tfrac{-5}{2\Delta x^2}+\tfrac{-5}{2\Delta y^2}+\tfrac{-5}{2\Delta z^2})U(t,\vec{X}) \\
+ \;& \tfrac{4}{3\Delta x^2}(U(t,x+\Delta x,y,z)+U(t,x-\Delta x,y,z)) \\
+ \;& \tfrac{4}{3\Delta y^2}(U(t,x,y+\Delta y,z)+U(t,x,y-\Delta y,z)) \\
+ \;& \tfrac{4}{3\Delta z^2}(U(t,x,y,z+\Delta z)+U(t,x,y,z-\Delta z)) \\
- \;& \tfrac{1}{12\Delta x^2}(U(t,x+2\Delta x,y,z)+U(t,x-2\Delta x,y,z)) \\
- \;& \tfrac{1}{12\Delta y^2}(U(t,x,y+2\Delta y,z)+U(t,x,y-2\Delta y,z)) \\
- \;& \tfrac{1}{12\Delta z^2}(U(t,x,y,z+2\Delta z)+U(t,x,y,z-2\Delta z))
\end{aligned}
$$

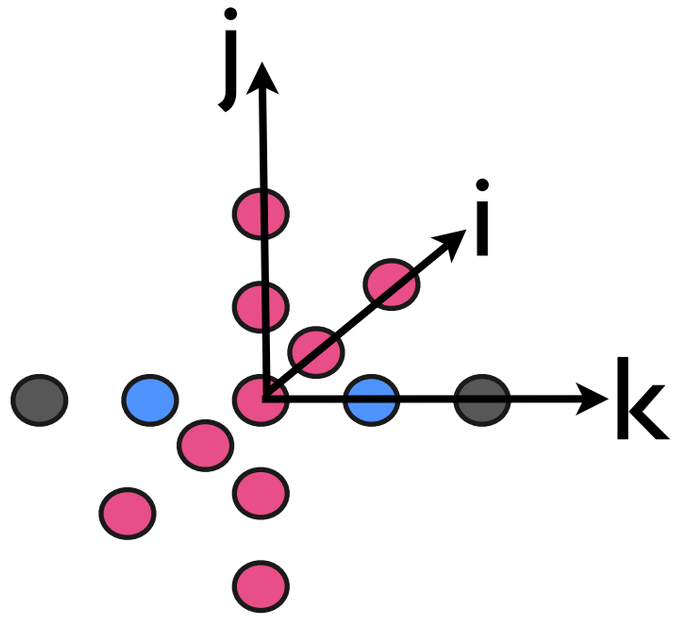
$$
\begin{aligned}
U\_in[i][j][k] &= U(t, i*\Delta x, j*\Delta y, k*\Delta z) \\
U\_out[i][j][k] &= U(t-\Delta t, i*\Delta x, j*\Delta y, k*\Delta z)
\end{aligned}
$$

$$
\begin{aligned}
c[0] &= \Delta t^2(\tfrac{-5}{2\Delta x^2}+\tfrac{-5}{2\Delta y^2}+\tfrac{-5}{2\Delta z^2})+2 \\
c[1] &= \tfrac{4\Delta t^2}{3\Delta x^2}, c[2]=\tfrac{4\Delta t^2}{3\Delta y^2}, c[3]=\tfrac{4\Delta t^2}{3\Delta z^2} \\
c[4] &= \tfrac{-\Delta t^2}{12\Delta x^2}, c[5]=\tfrac{-\Delta t^2}{12\Delta y^2}, c[6]=\tfrac{-\Delta t^2}{12\Delta z^2}
\end{aligned}
$$

```
U_out[k][j][i] = -U_out[k][j][i]
                 + c[0]*U_in[k][j][i]
                 + c[3]*(U_in[k+1][j][i]+U_in[k-1][j][i])
                 + c[2]*(U_in[k][j+1][i]+U_in[k][j-1][i])
                 + c[1]*(U_in[k][j][i+1]+U_in[k][j][i-1])
                 + c[6]*(U_in[k+2][j][i]+U_in[k-2][j][i])
                 + c[5]*(U_in[k][j+2][i]+U_in[k][j-2][i])
                 + c[4]*(U_in[k][j][i+2]+U_in[k][j][i-2]);
```

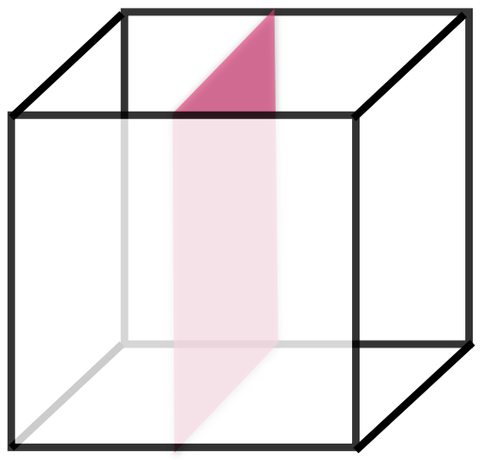
Stencil sweep in terms of planes

```
for(k = start;k <= end;k++)
      for(j = 1;j < ny;j++)
            for(i = 1;i < nx; i++)
                U_out[k][j][i] = -U_out[k][j][i]
                + c[0]*U_in[k][j][i]
                + c[3]*(U_in[k+1][j][i]+U_in[k-1][j][i])
                + c[2]*(U_in[k][j+1][i]+U_in[k][j-1][i])
                + c[1]*(U_in[k][j][i+1]+U_in[k][j][i-1])
                + c[6]*(U_in[k+2][j][i]+U_in[k-2][j][i])
                + c[5]*(U_in[k][j+2][i]+U_in[k][j-2][i])
                + c[4]*(U_in[k][j][i+2]+U_in[k][j][i-2]);
```

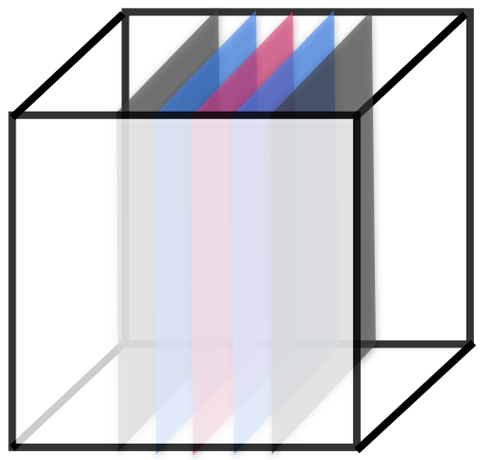


Stencil Order = 2

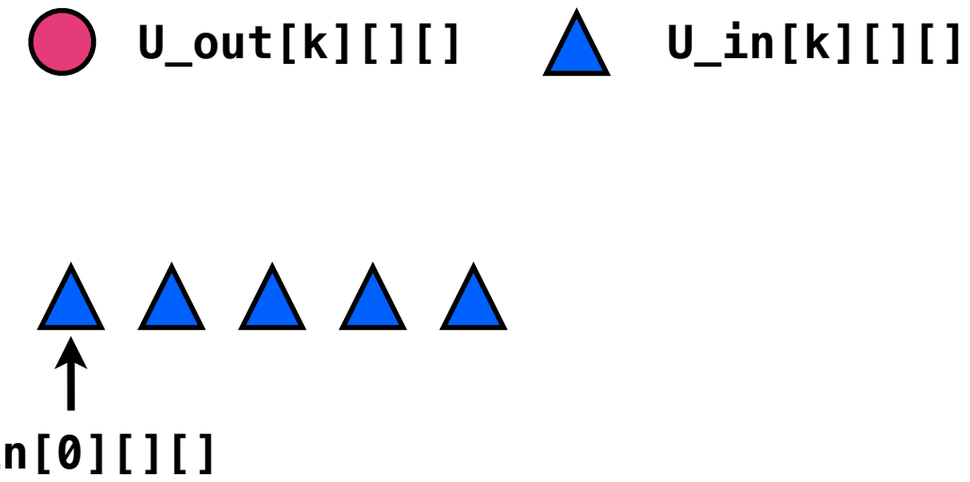
`U_out[k][][]`



`U_in[k-2:k+2][][]`

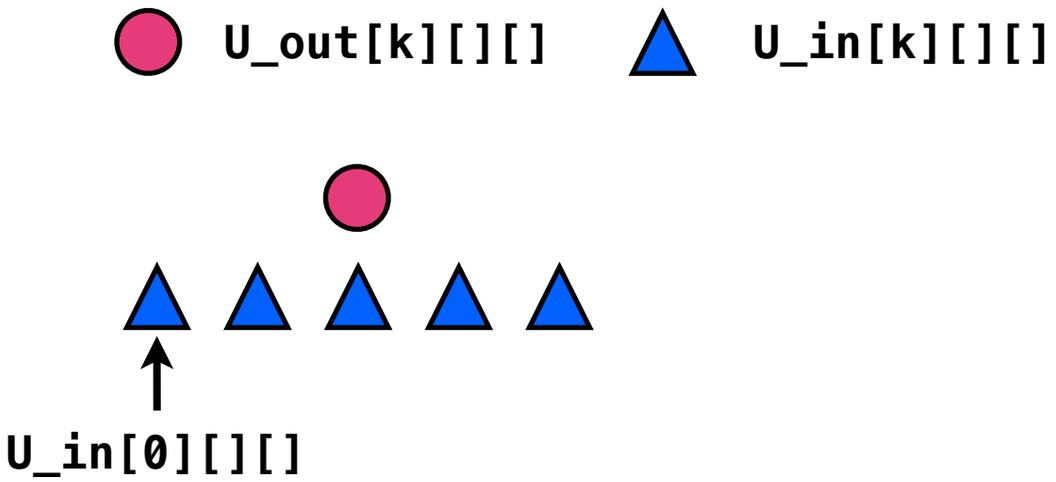


Naive Implementation



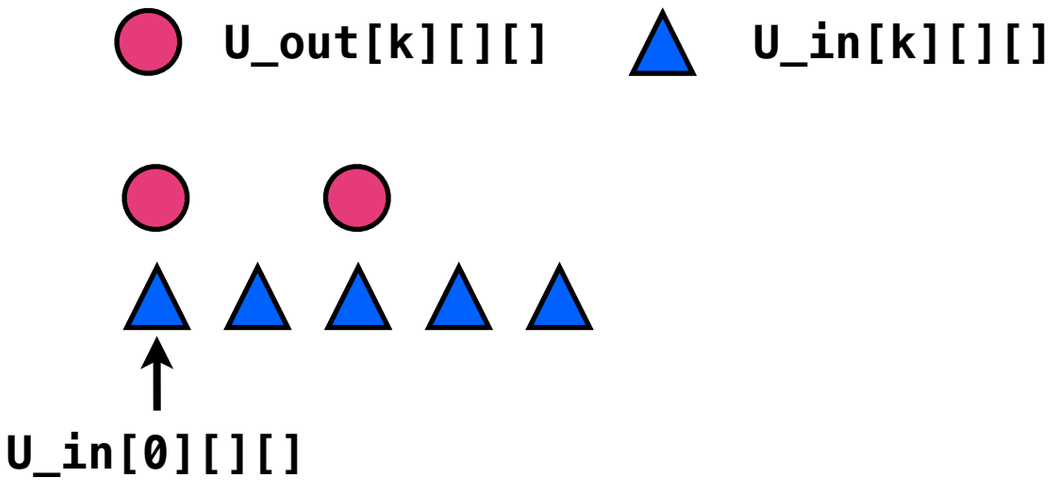
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



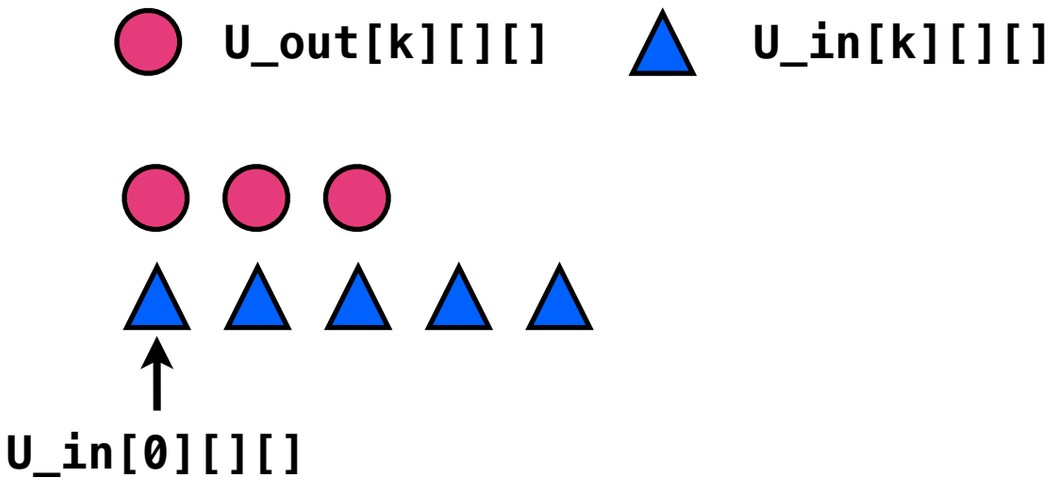
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



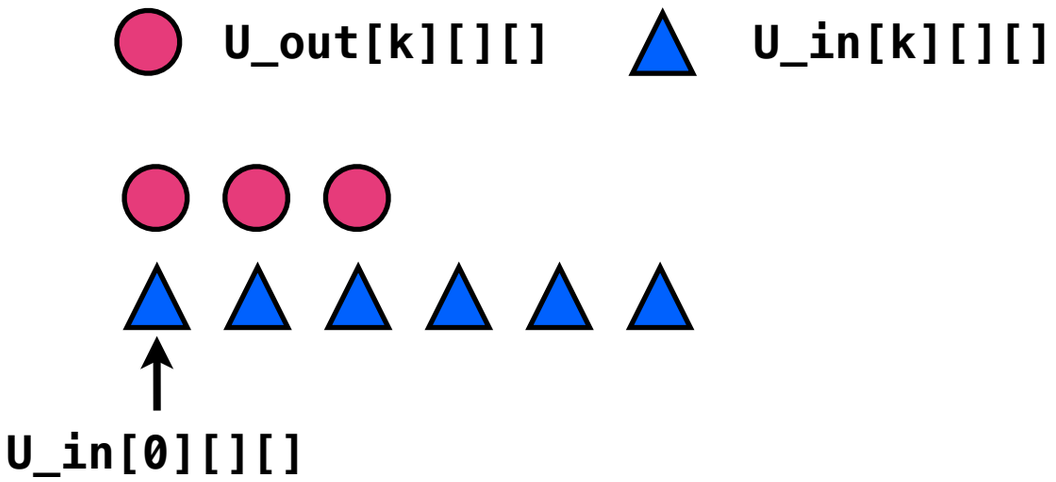
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



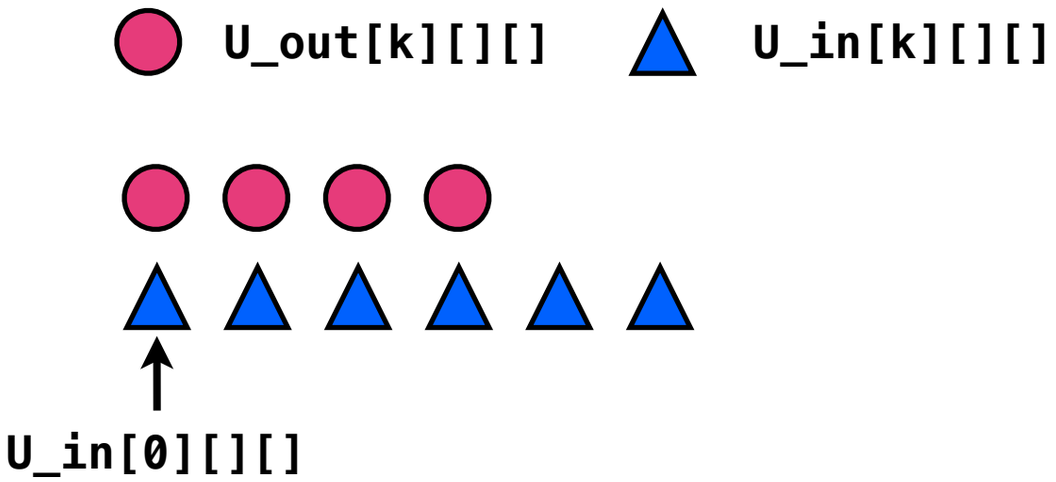
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



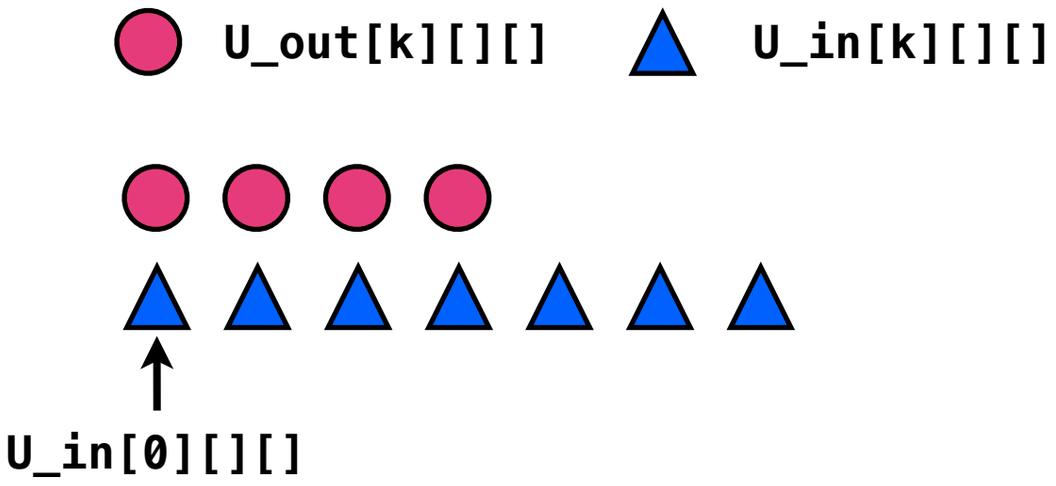
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



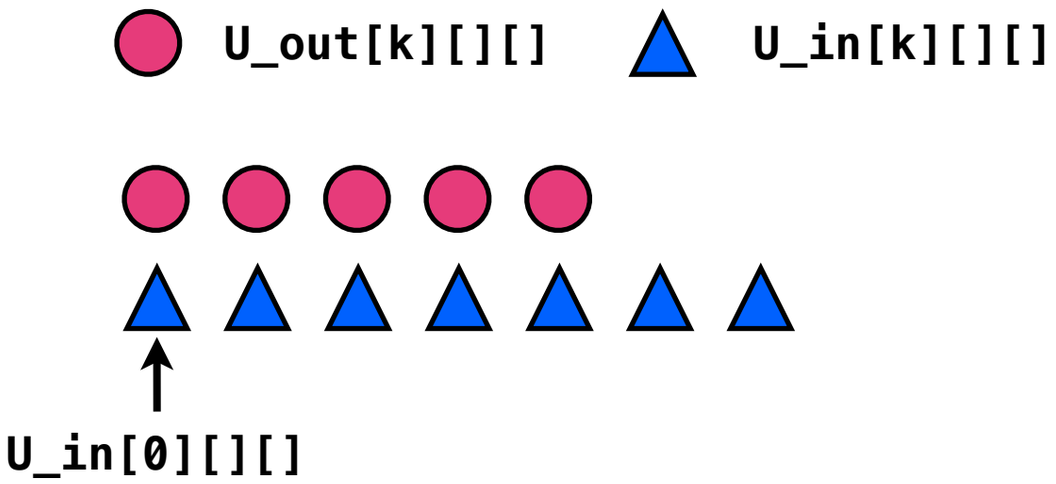
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



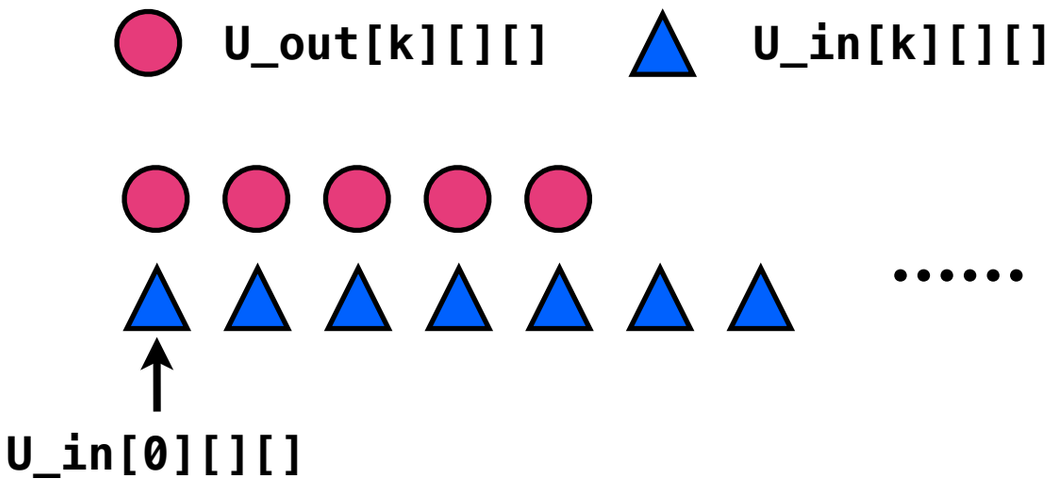
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



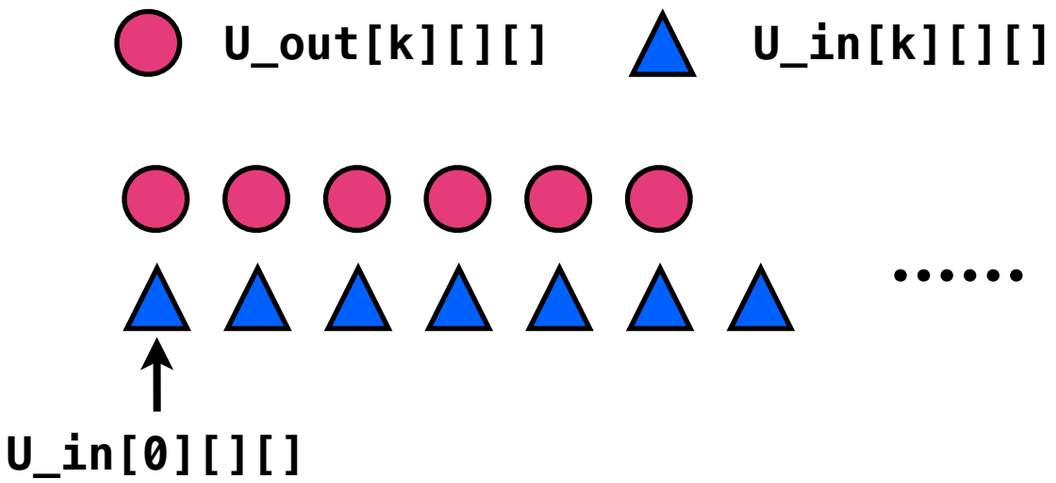
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



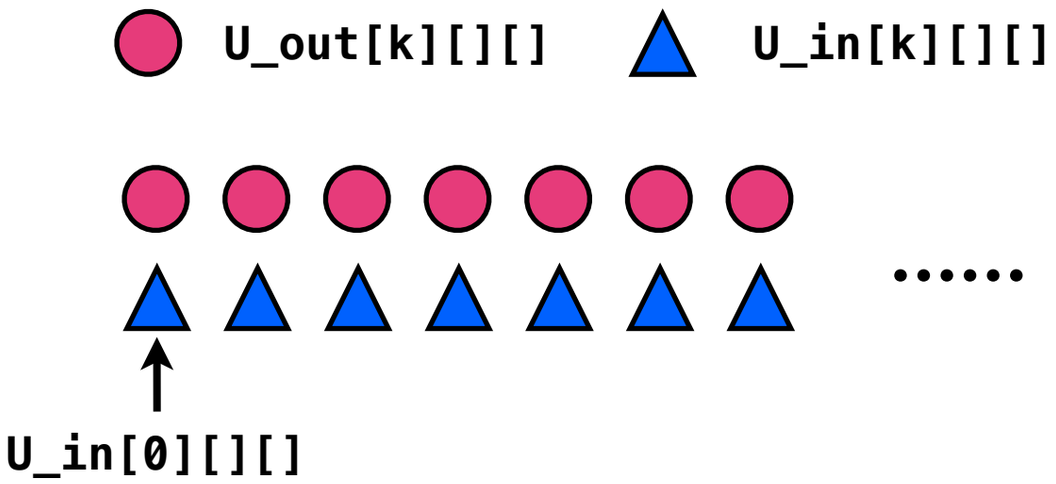
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



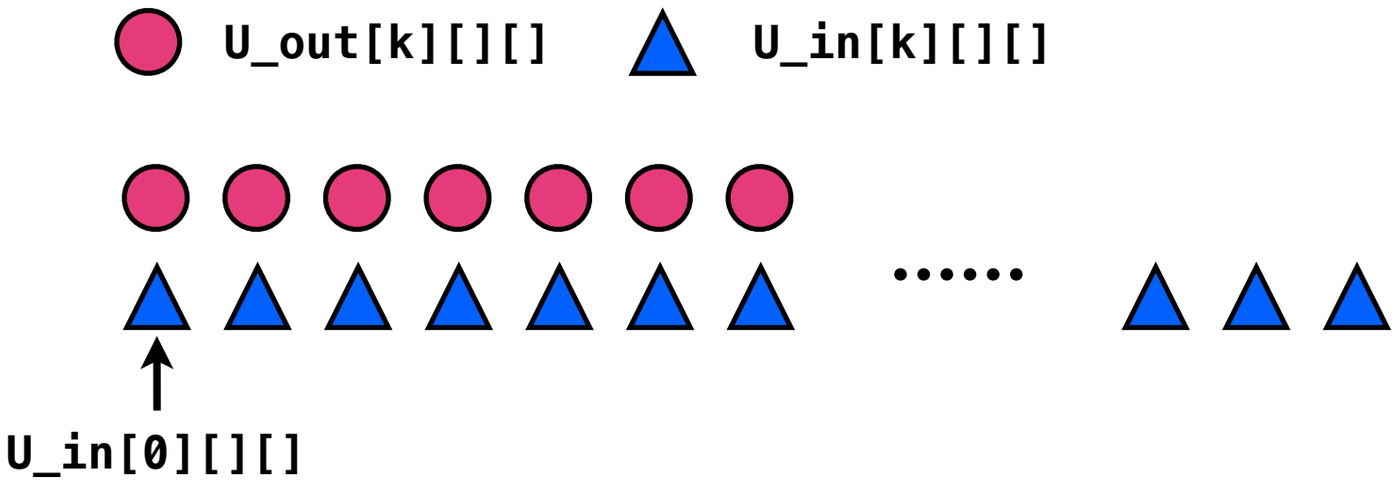
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



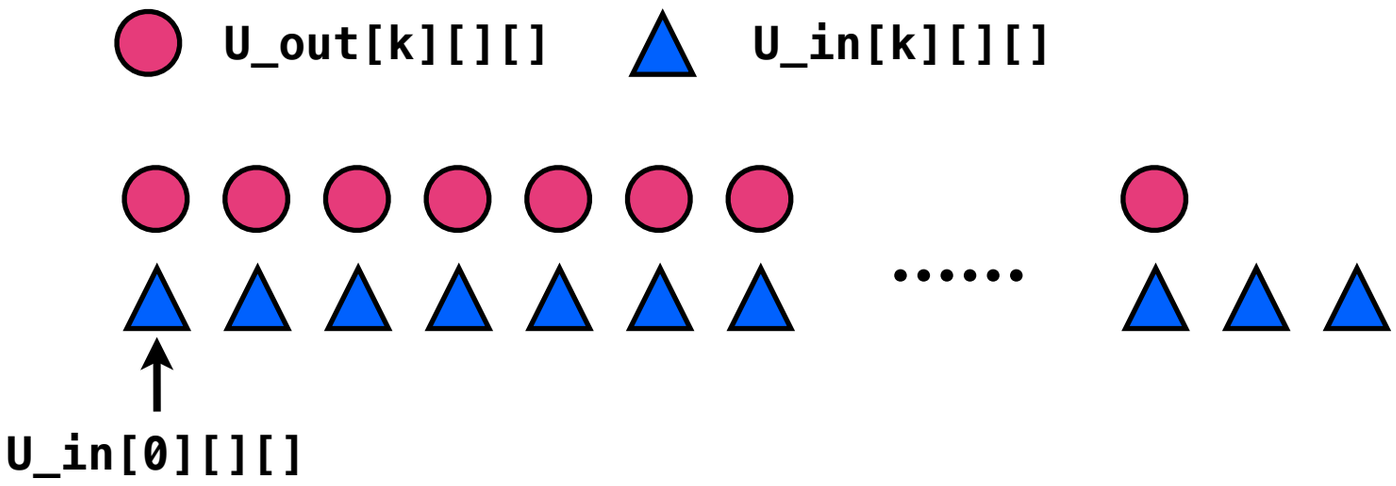
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



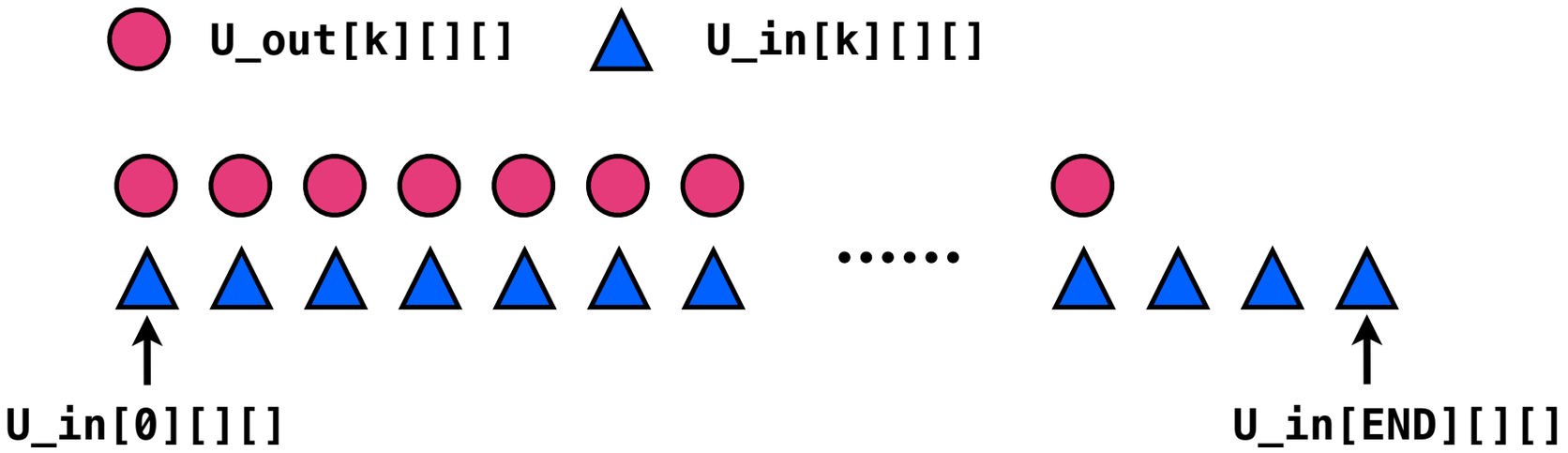
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



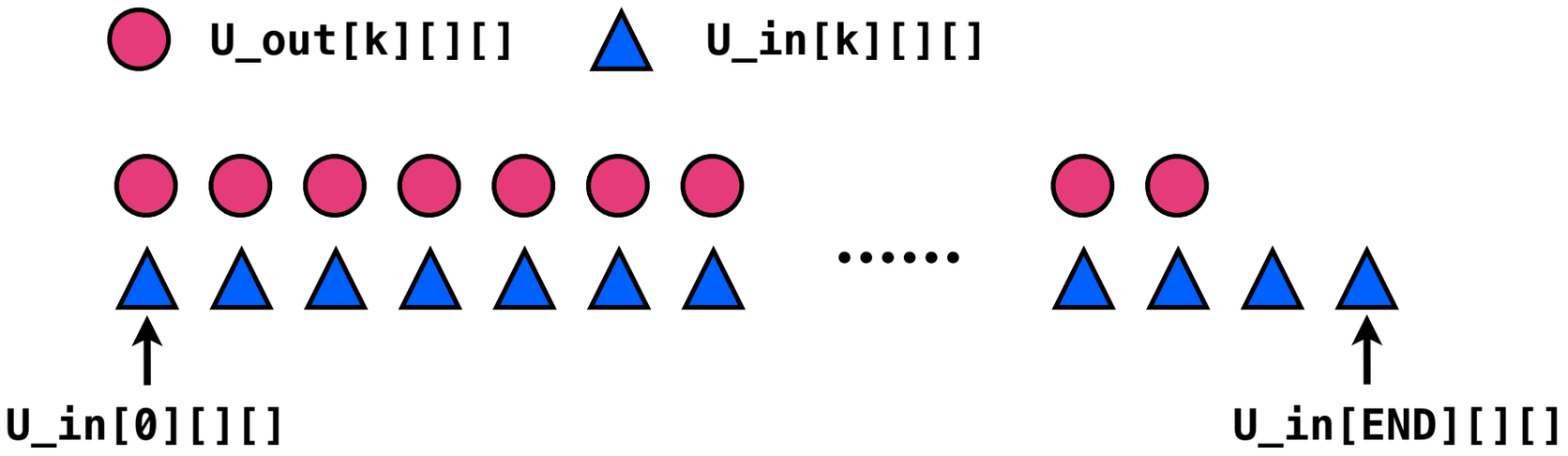
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



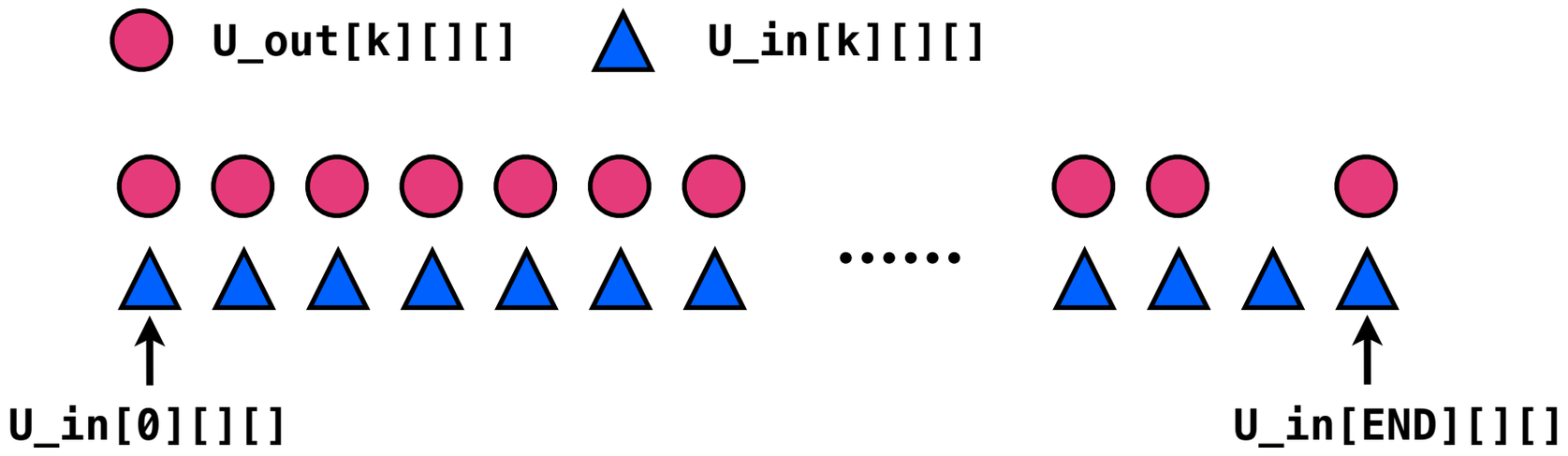
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



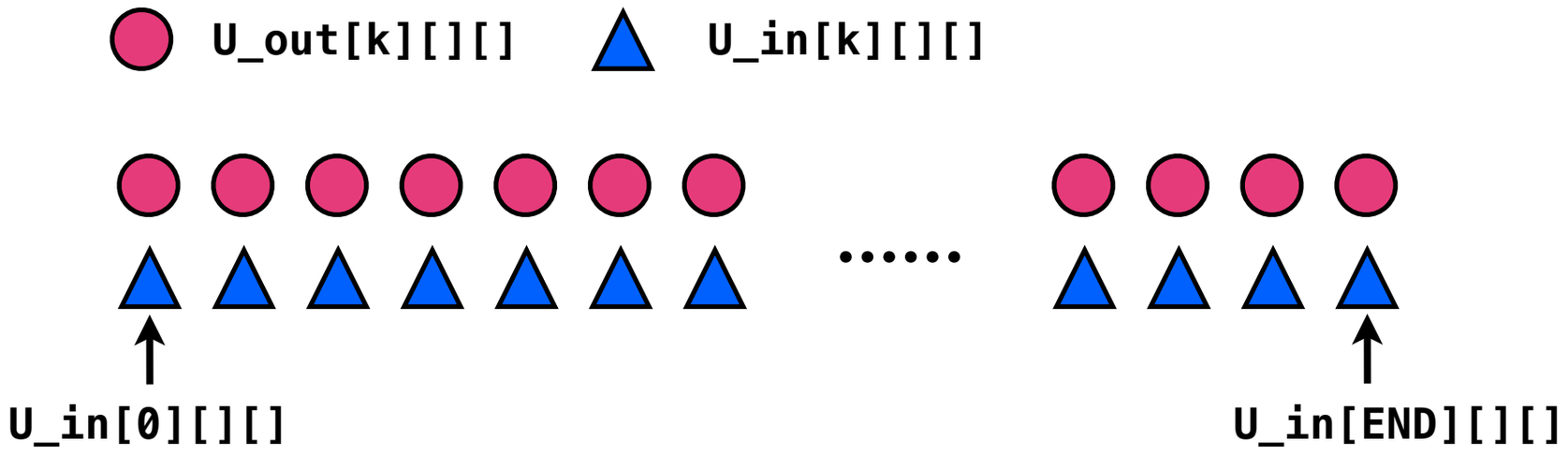
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



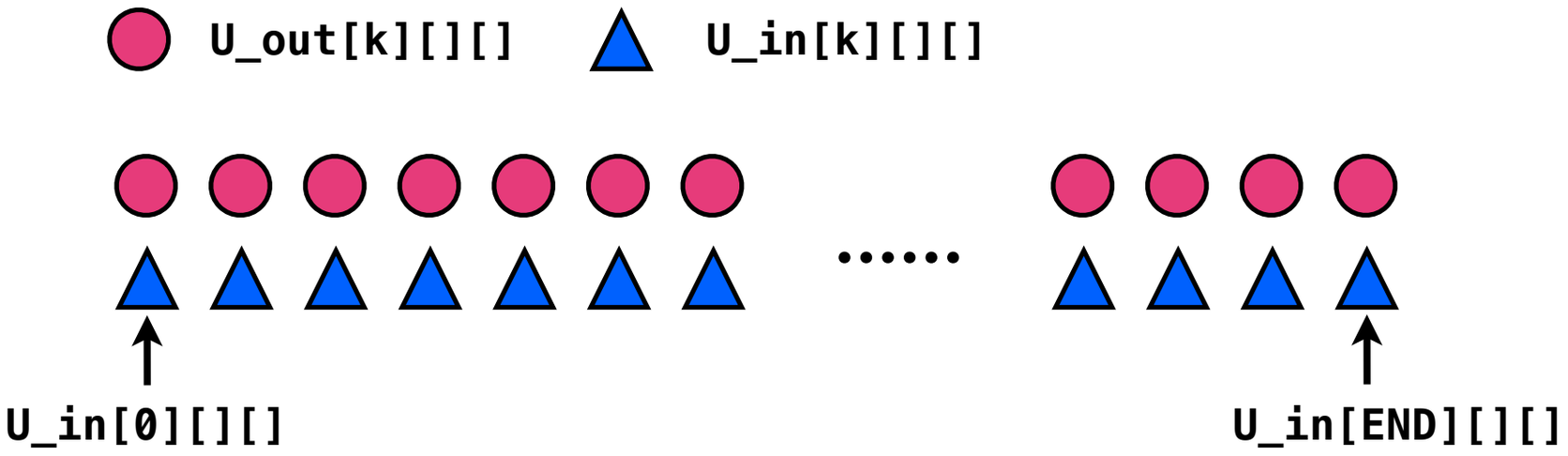
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation



http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

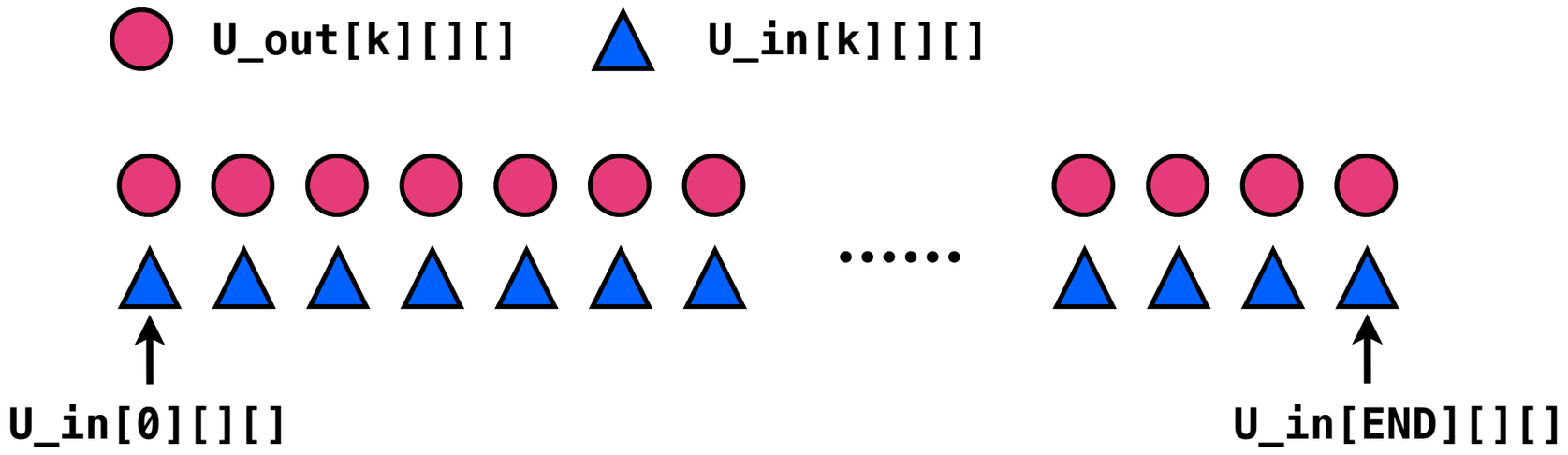
Naive Implementation



How can I further speed up the code?

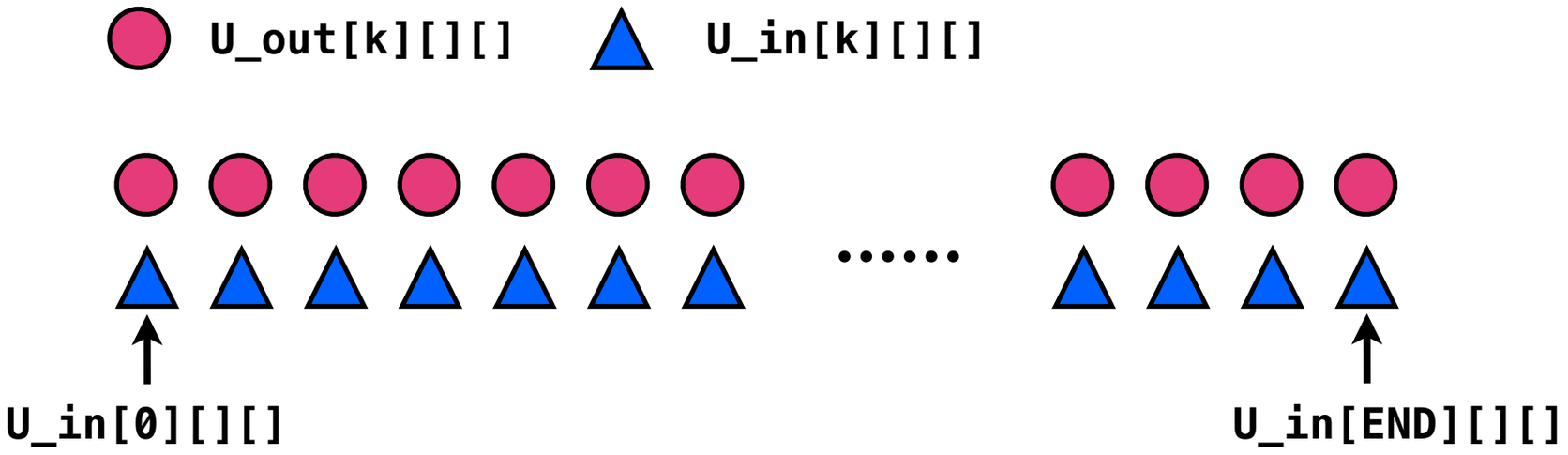
http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

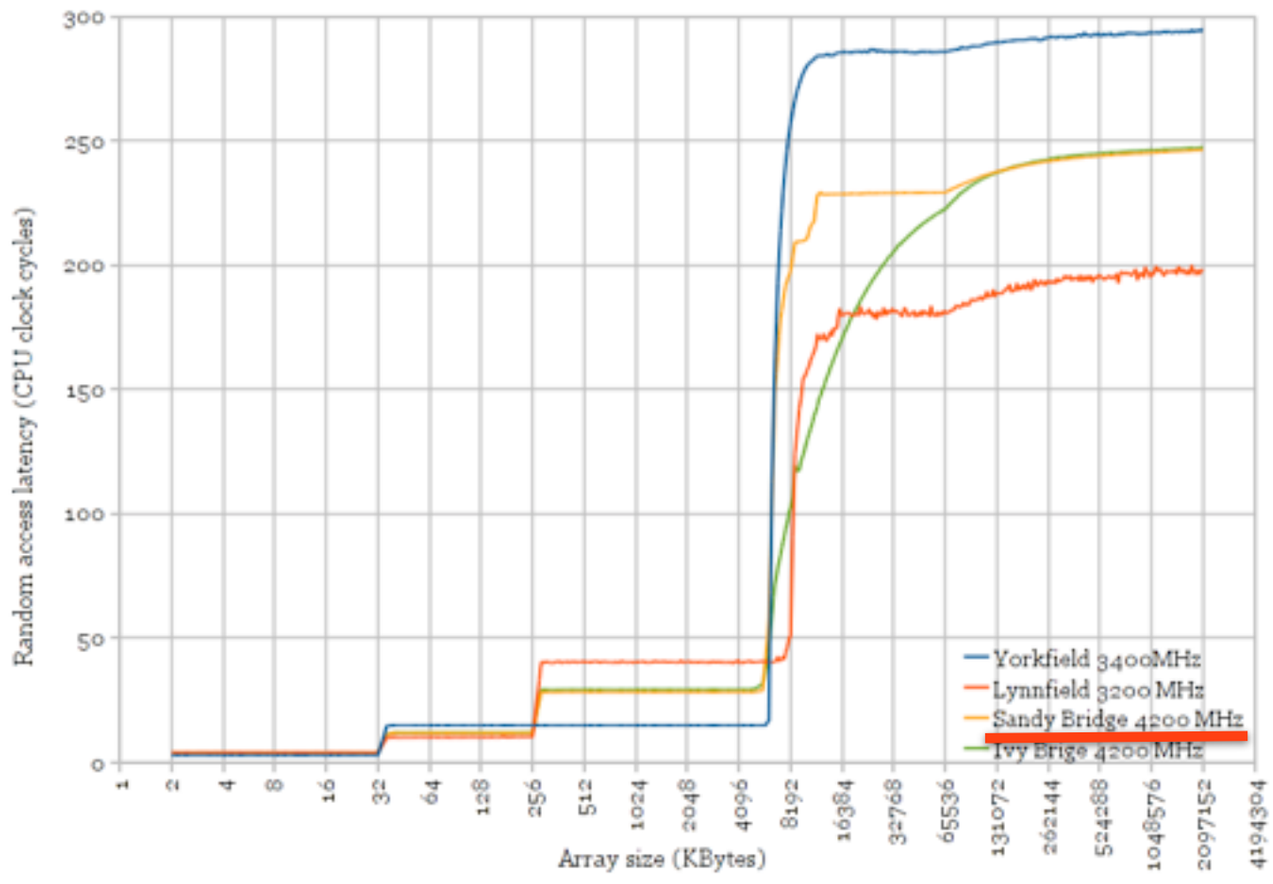
Naive Implementation



http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Naive Implementation





http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

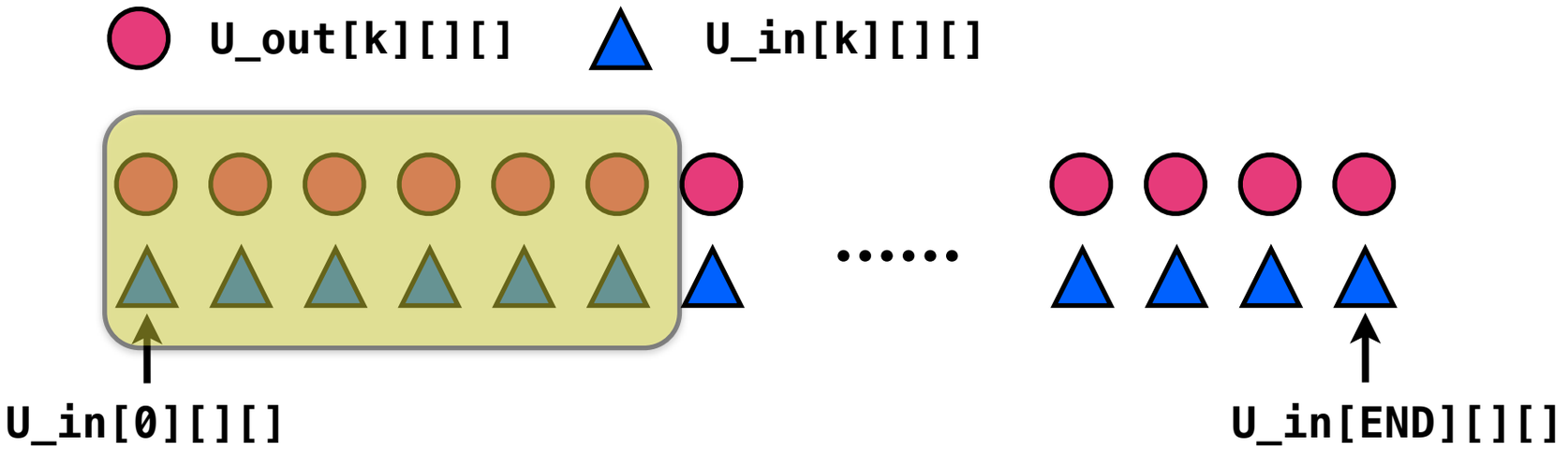
Accessing Time: L1, L2, L3 < 50 CPU cycles
RAM 200~250 CPU cycles

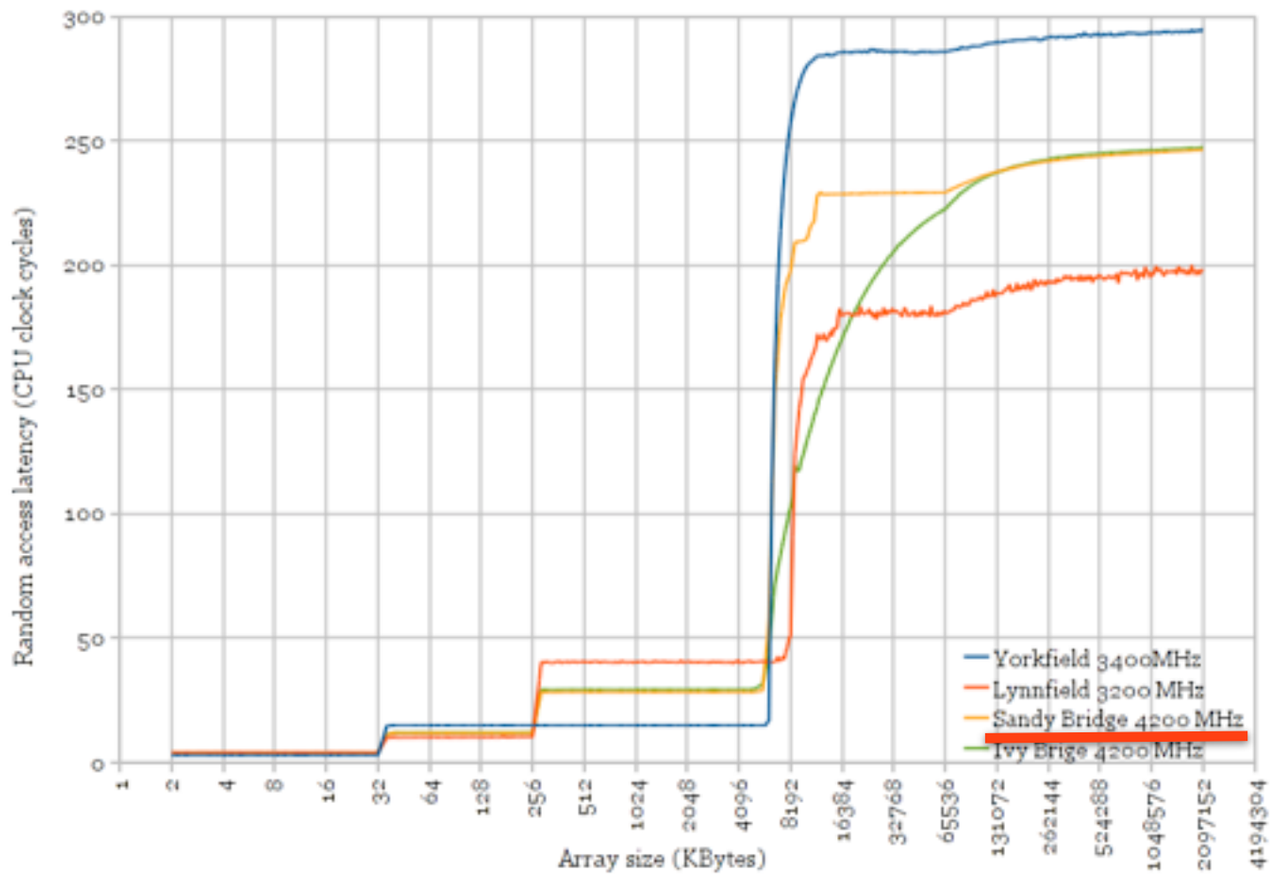
Cache Replacement Policy (FIFO, LFU)
L3 cache size is 12MB* shared by 6 cores in one Westmere node(DAVinCI, Rice U)

Consider the array size is 260*260*260
12*10^6/4/(260*260) = 44.379 (using float)
It implies that at most 22 planes of each data array can be inserted into the cache.

* http://www.cpu-world.com/sspec/Q4/Q4EN.html

Naive Implementation





http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

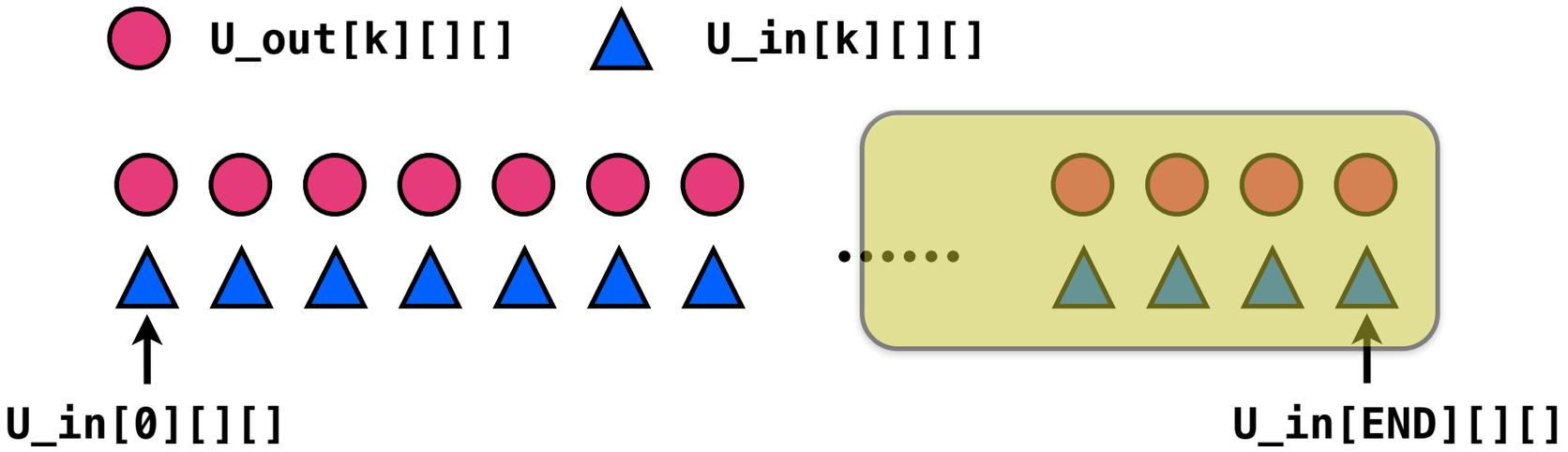
Accessing Time: L1, L2, L3 < 50 CPU cycles
RAM 200~250 CPU cycles

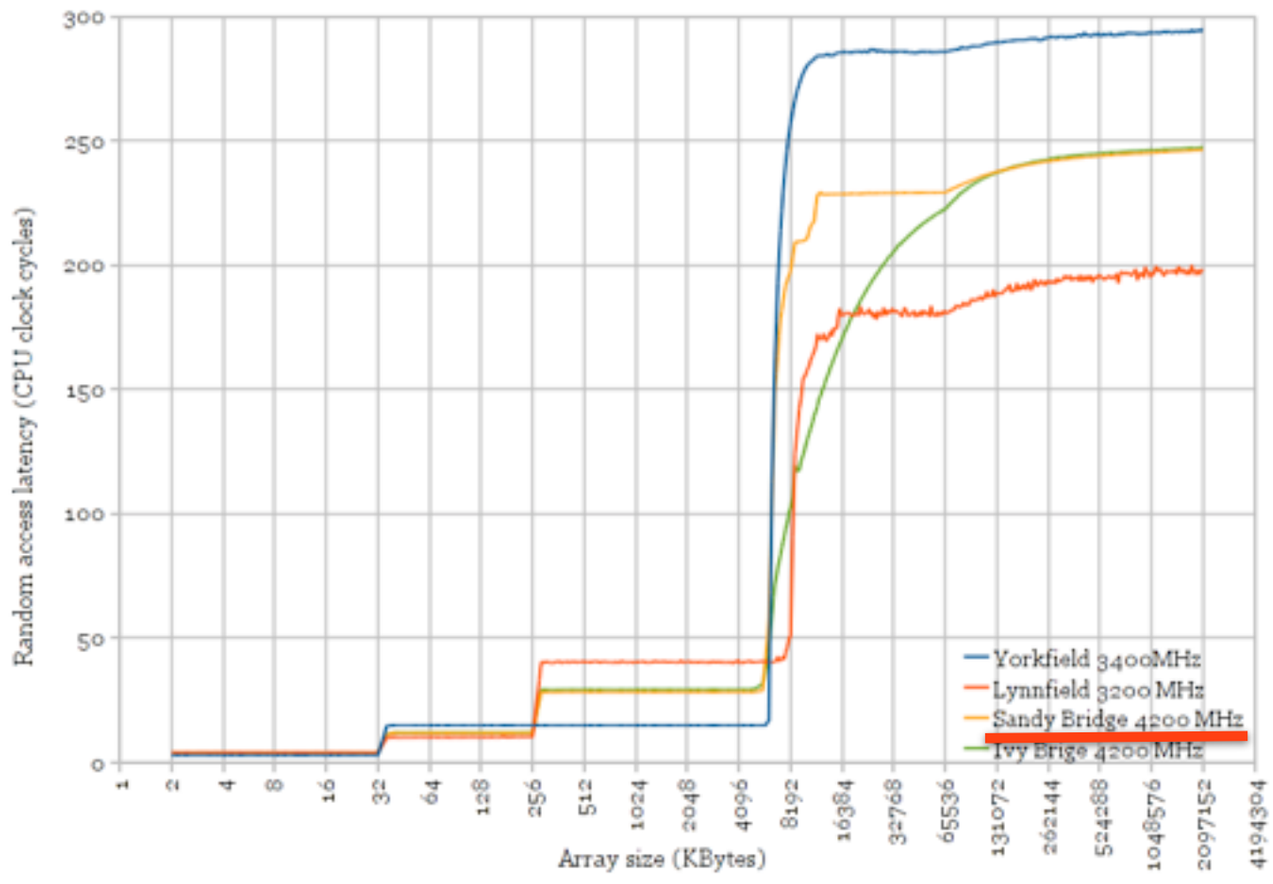
Cache Replacement Policy (FIFO, LFU)
L3 cache size is 12MB* shared by 6 cores in one Westmere node(DAVinCI, Rice U)

Consider the array size is 260*260*260
12*10^6/4/(260*260) = 44.379 (using float)
It implies that at most 22 planes of each data array can be inserted into the cache.

* http://www.cpu-world.com/sspec/Q4/Q4EN.html

Naive Implementation





http://blog.stuffedcow.net/2013/01/ivb-cache-replacement/

Accessing Time: L1, L2, L3 < 50 CPU cycles
RAM 200~250 CPU cycles

Cache Replacement Policy (FIFO, LFU)
L3 cache size is 12MB* shared by 6 cores in one Westmere node(DAVinCI, Rice U)

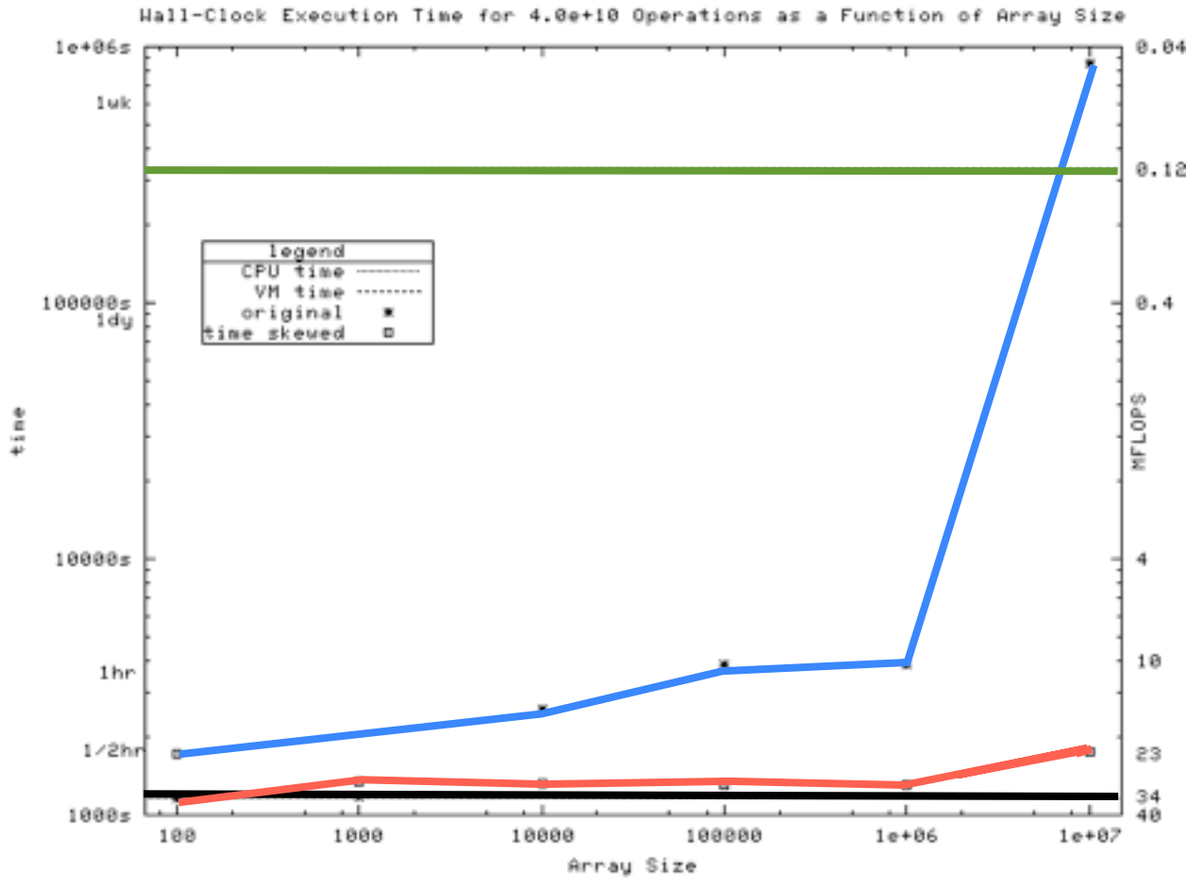
Consider the array size is 260*260*260
12*10^6/4/(260*260) = 44.379 (using float)
It implies that at most 22 planes of each data array can be inserted into the cache.

* http://www.cpu-world.com/sspec/Q4/Q4EN.html

The Time-Skewing Approach

Time skewing is an optimization that can be used to optimize certain regular scientific codes for machines vastly outperforms the memory systems.

http://www.haverford.edu/computerscience/faculty/davew/cache-opt/cache-opt.html



*In the acquisition design studies for subsalt imaging, the computational model has 640*640*400 grid points. For the vector acoustic method, 14 3D arrays(approximately 10 GB of addressable memory) is required.

* John T. Etgen, Michael J. OBrien, *Computational methods for large-scale 3D acoustic finite-difference modeling*: a tutorial. Geophysics 72 (2007) SM223SM230.

Domain Decomposition?

If high-speed communication among processors is possible, the approach has the merit.
Drawbacks: complicated language, increasing the probability of failure

Time-Skewing Approach Description(5 steps, tile size = 4)

`U_out[k][][]`

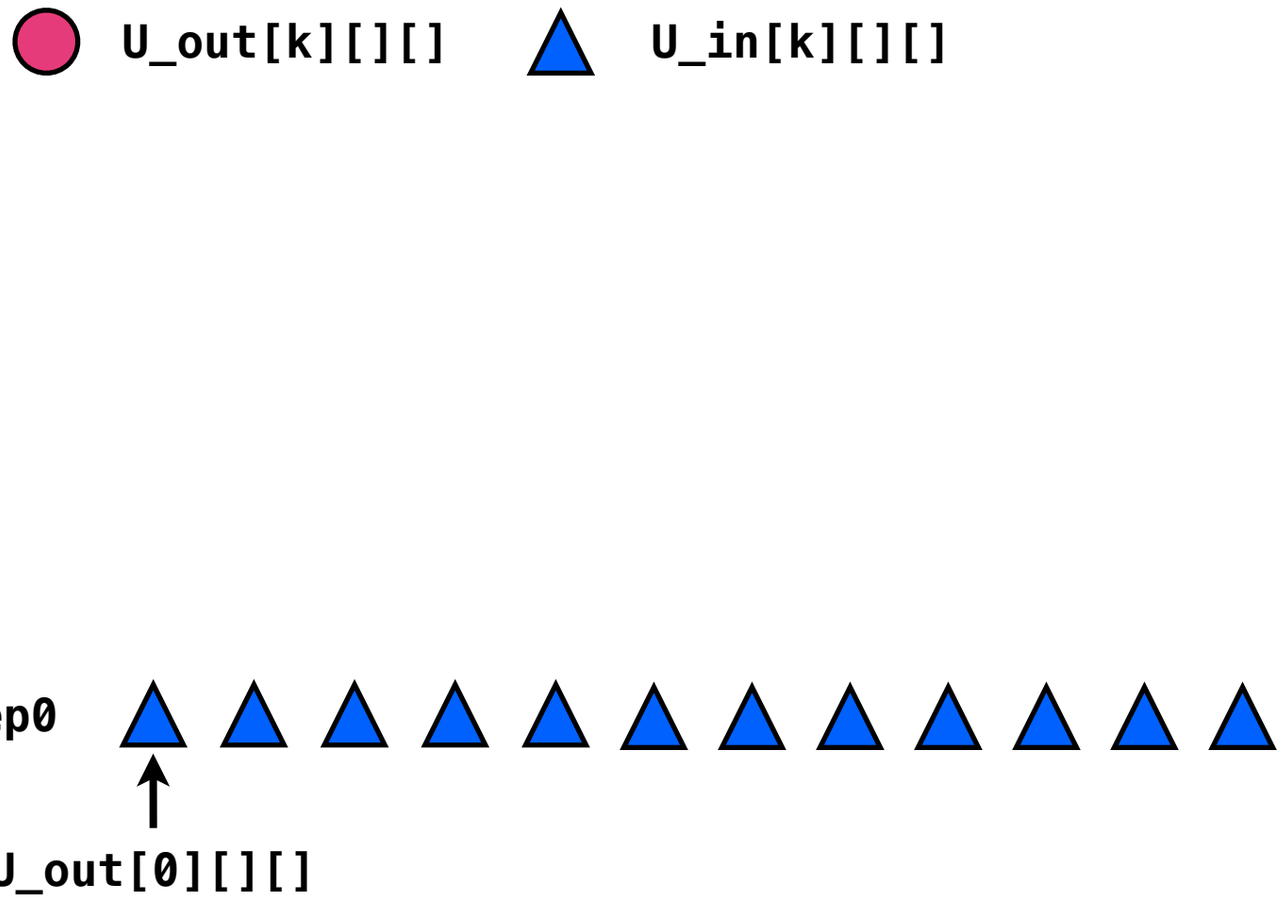
`U_in[k][][]`

Time-Skewing Approach Description(5 steps, tile size = 4)

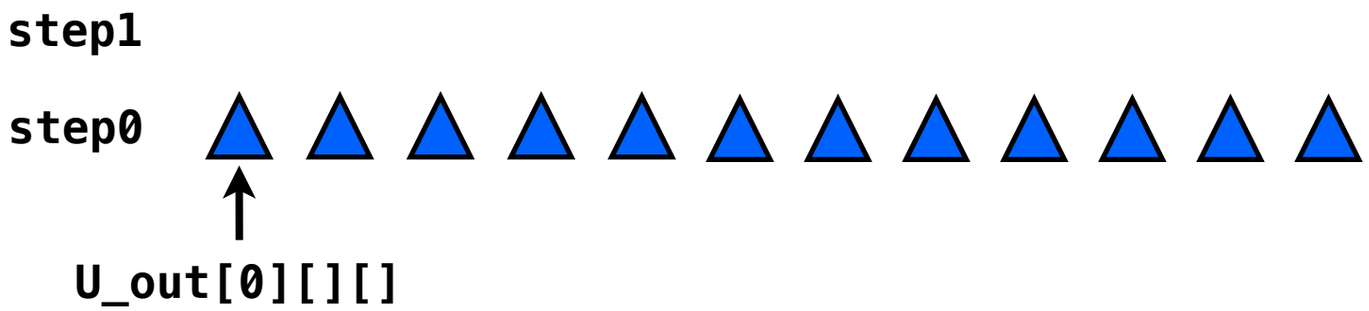

`U_out[k][][]` `U_in[k][][]`

`step0`

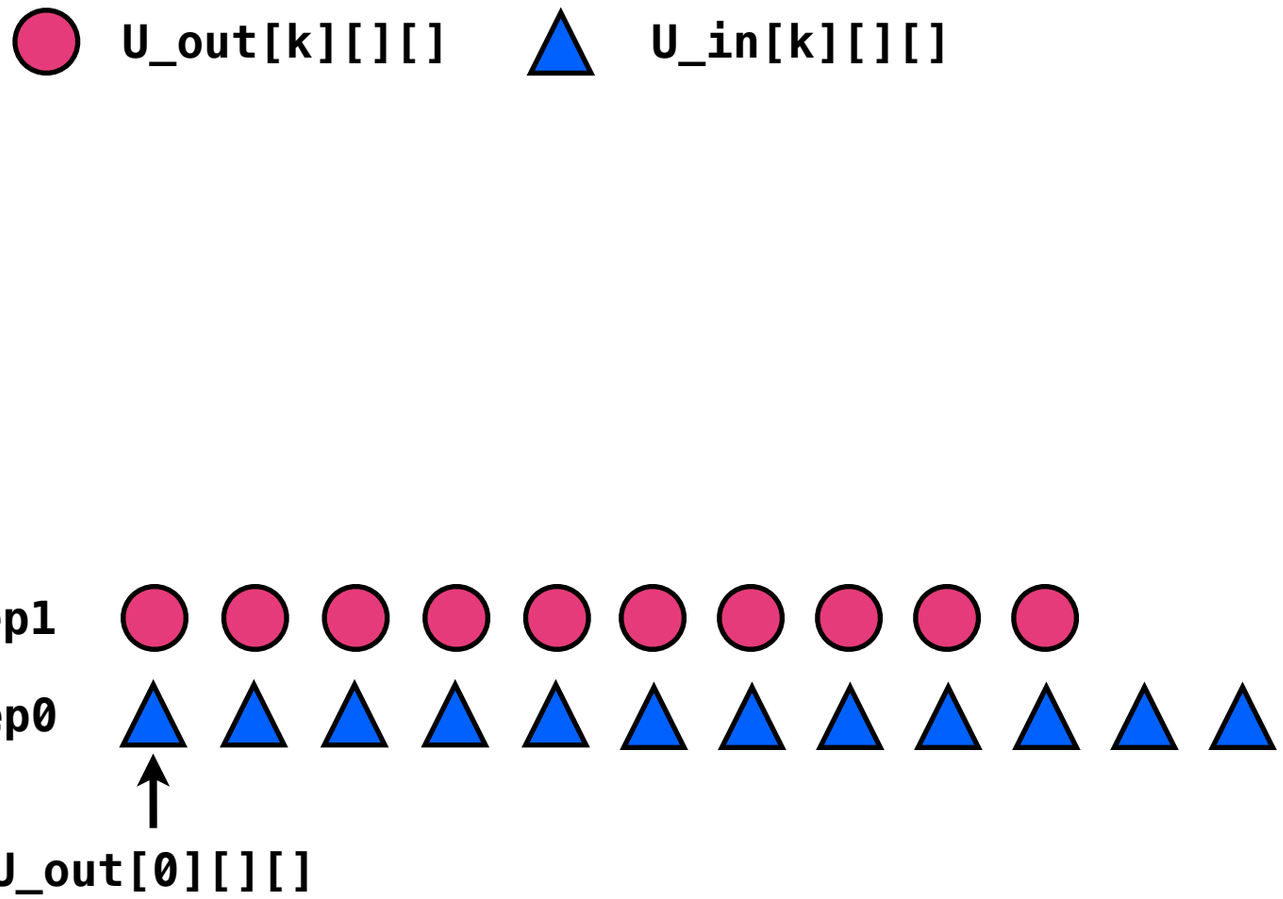
Time-Skewing Approach Description(5 steps, tile size = 4)



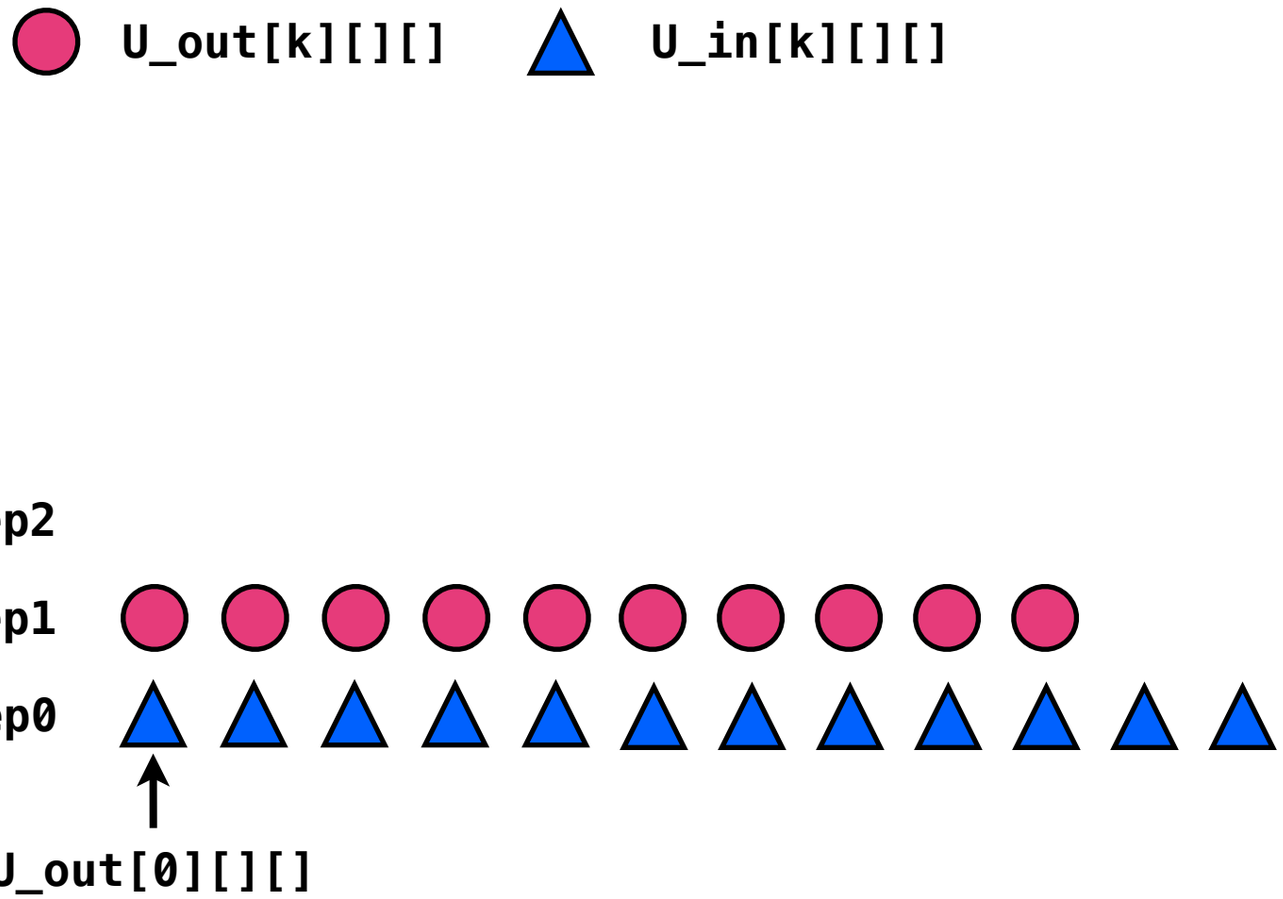
Time-Skewing Approach Description(5 steps, tile size = 4)



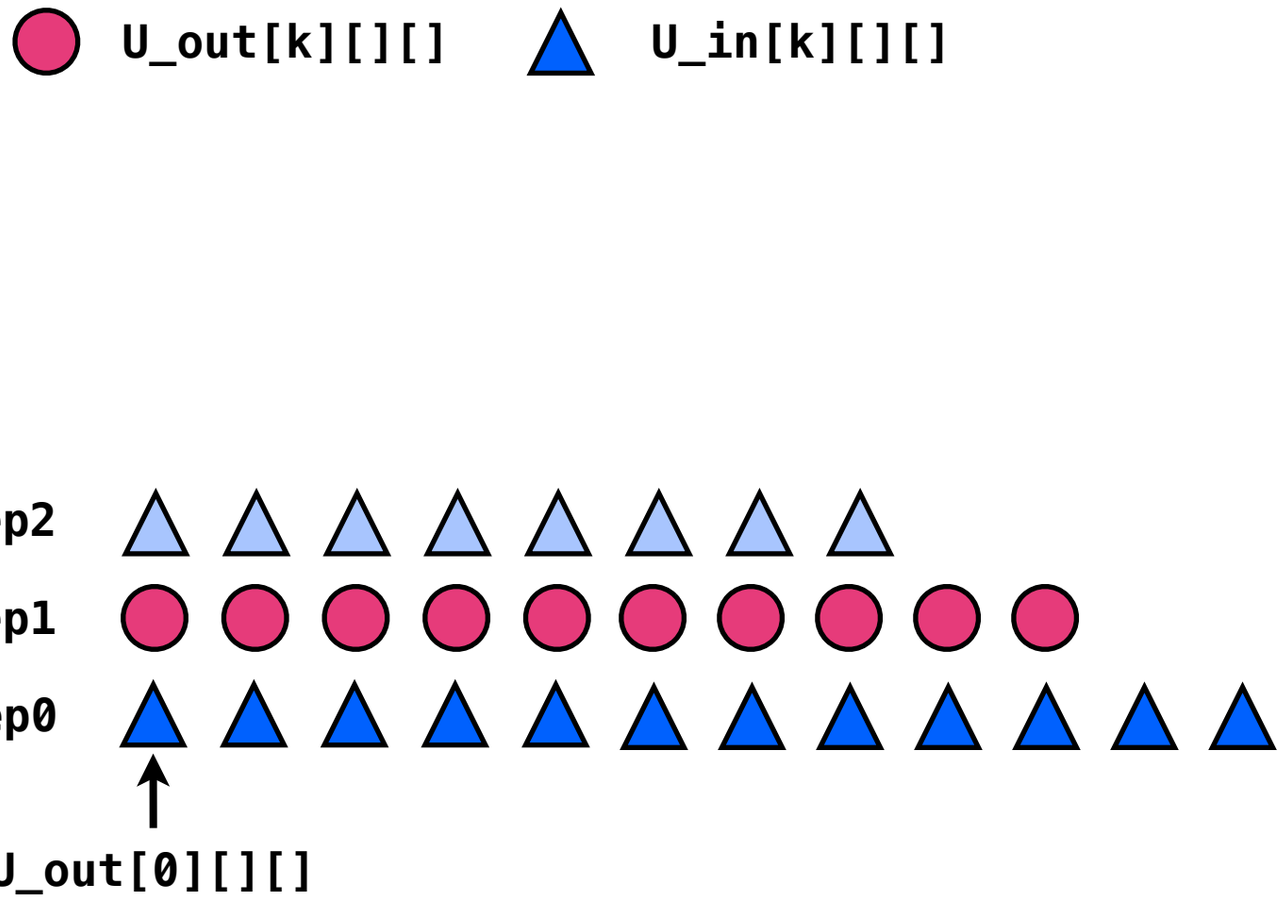
Time-Skewing Approach Description(5 steps, tile size = 4)



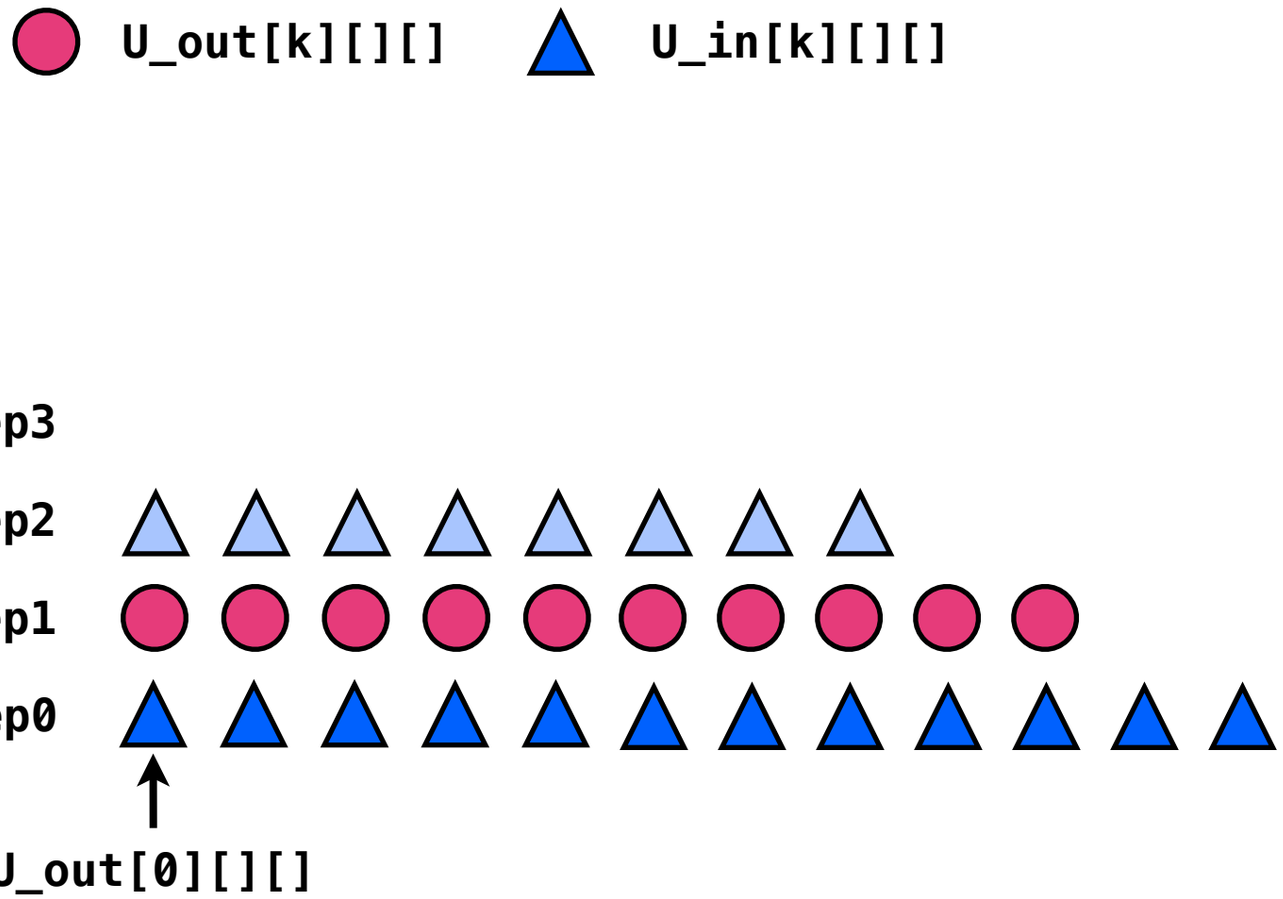
Time-Skewing Approach Description(5 steps, tile size = 4)



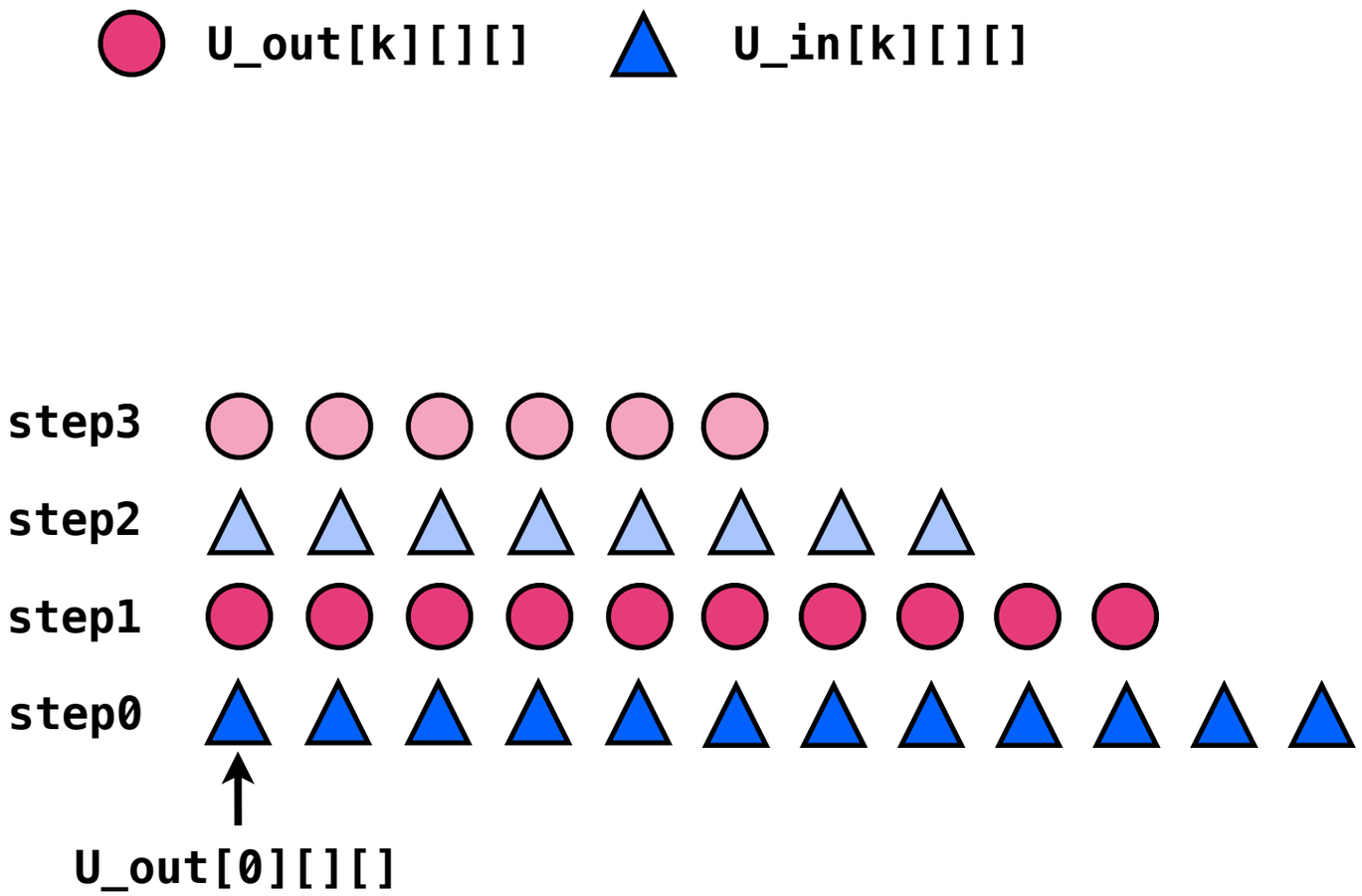
Time-Skewing Approach Description(5 steps, tile size = 4)



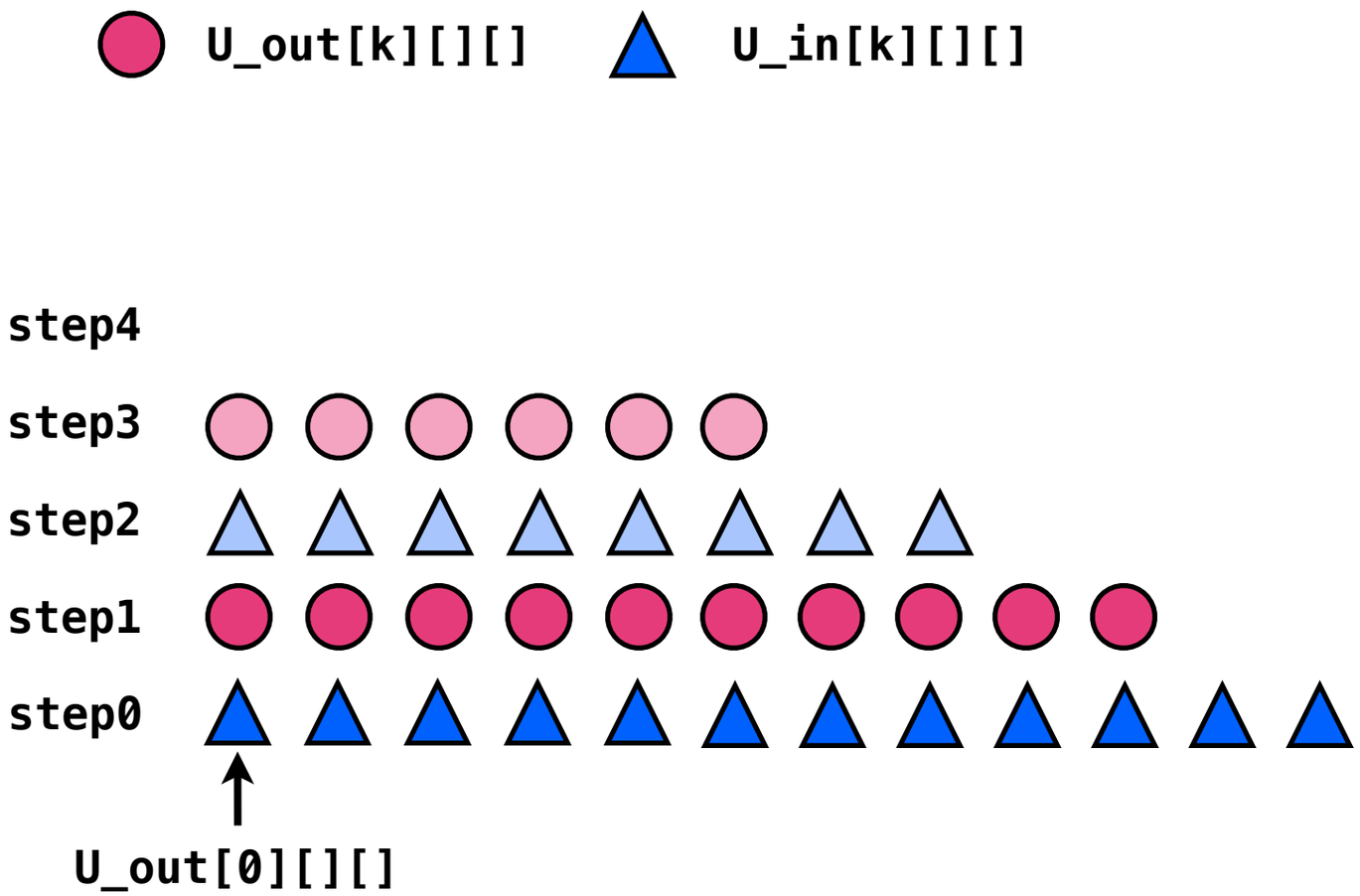
Time-Skewing Approach Description(5 steps, tile size = 4)



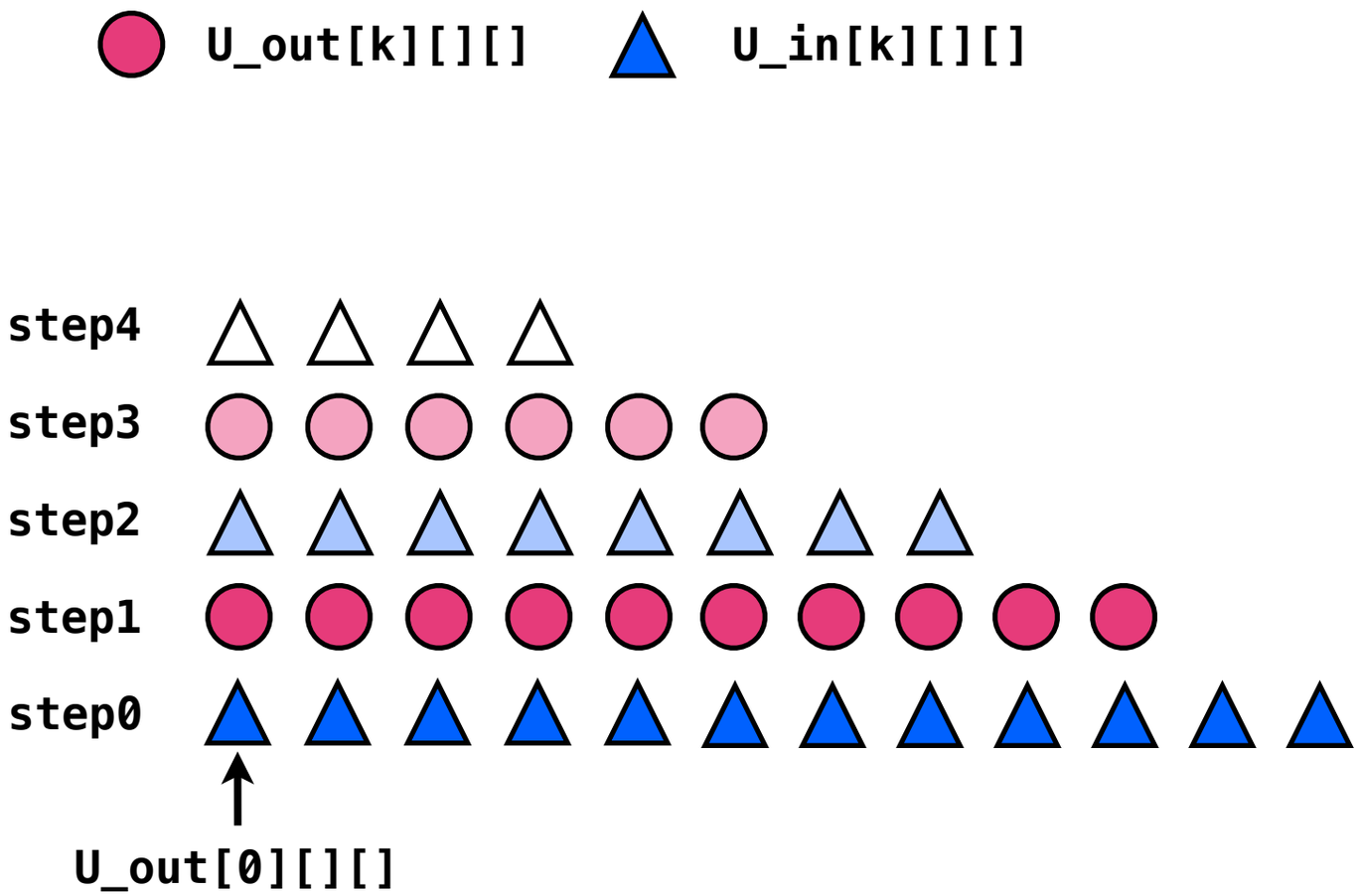
Time-Skewing Approach Description(5 steps, tile size = 4)



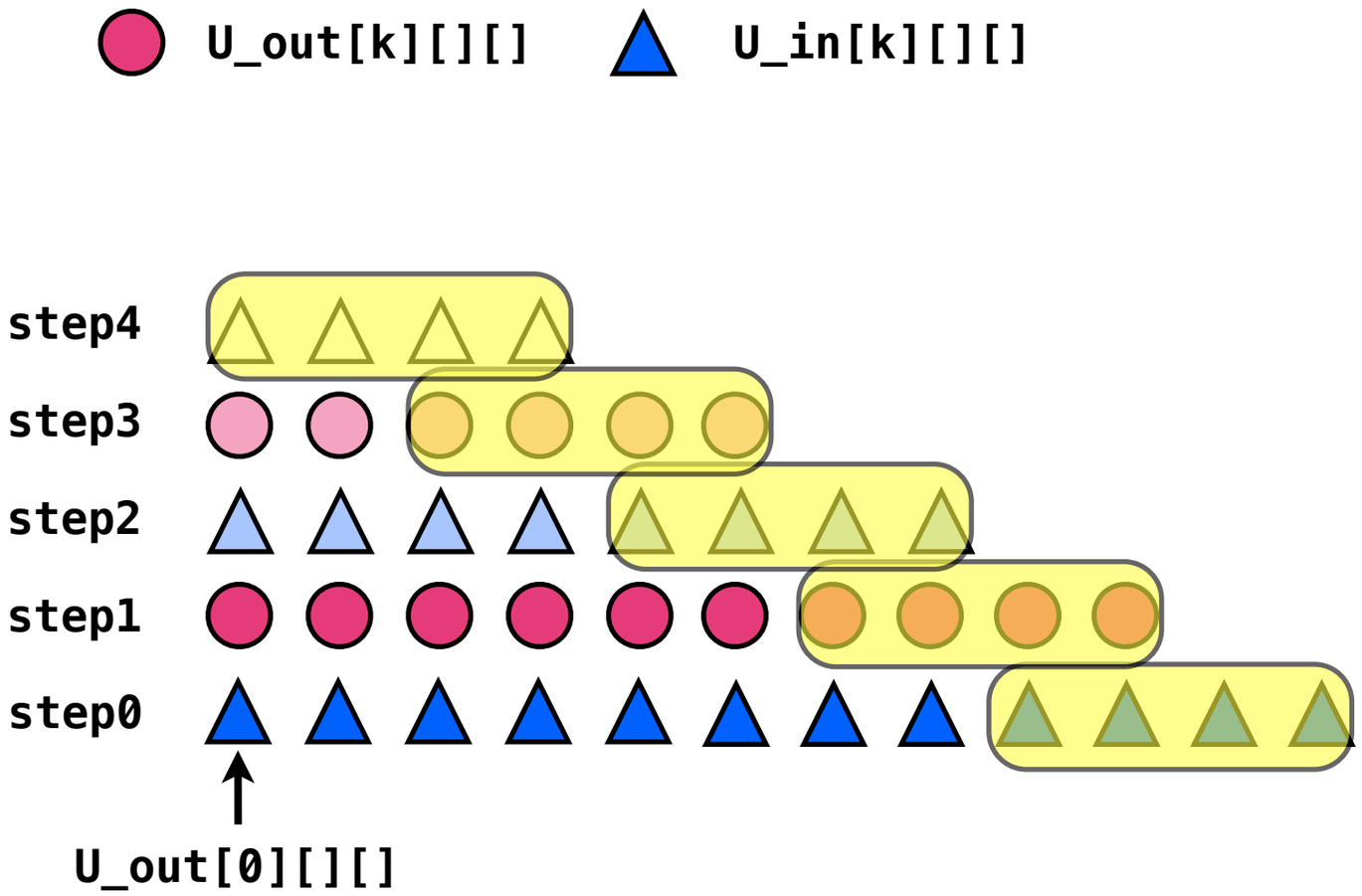
Time-Skewing Approach Description(5 steps, tile size = 4)



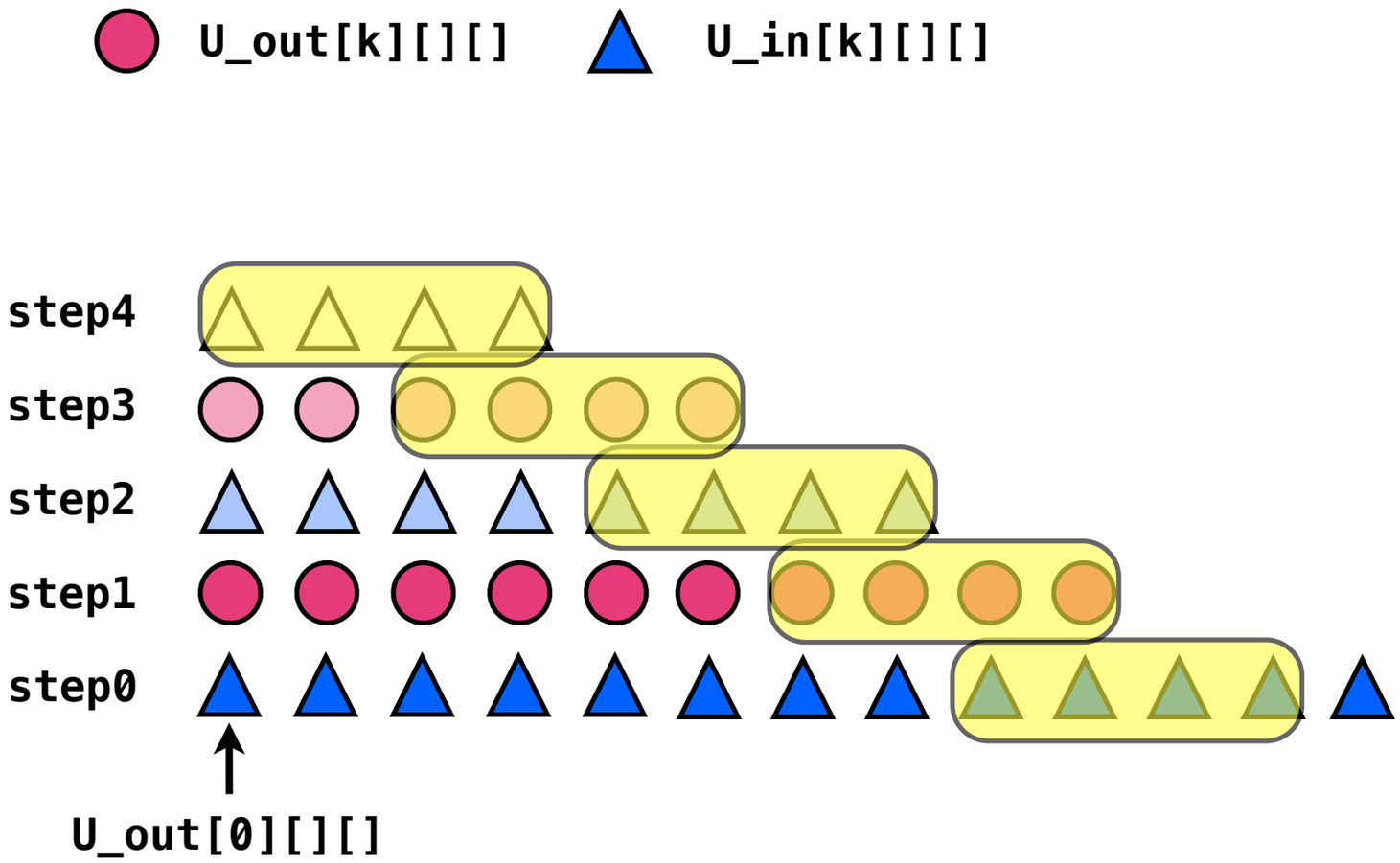
Time-Skewing Approach Description(5 steps, tile size = 4)



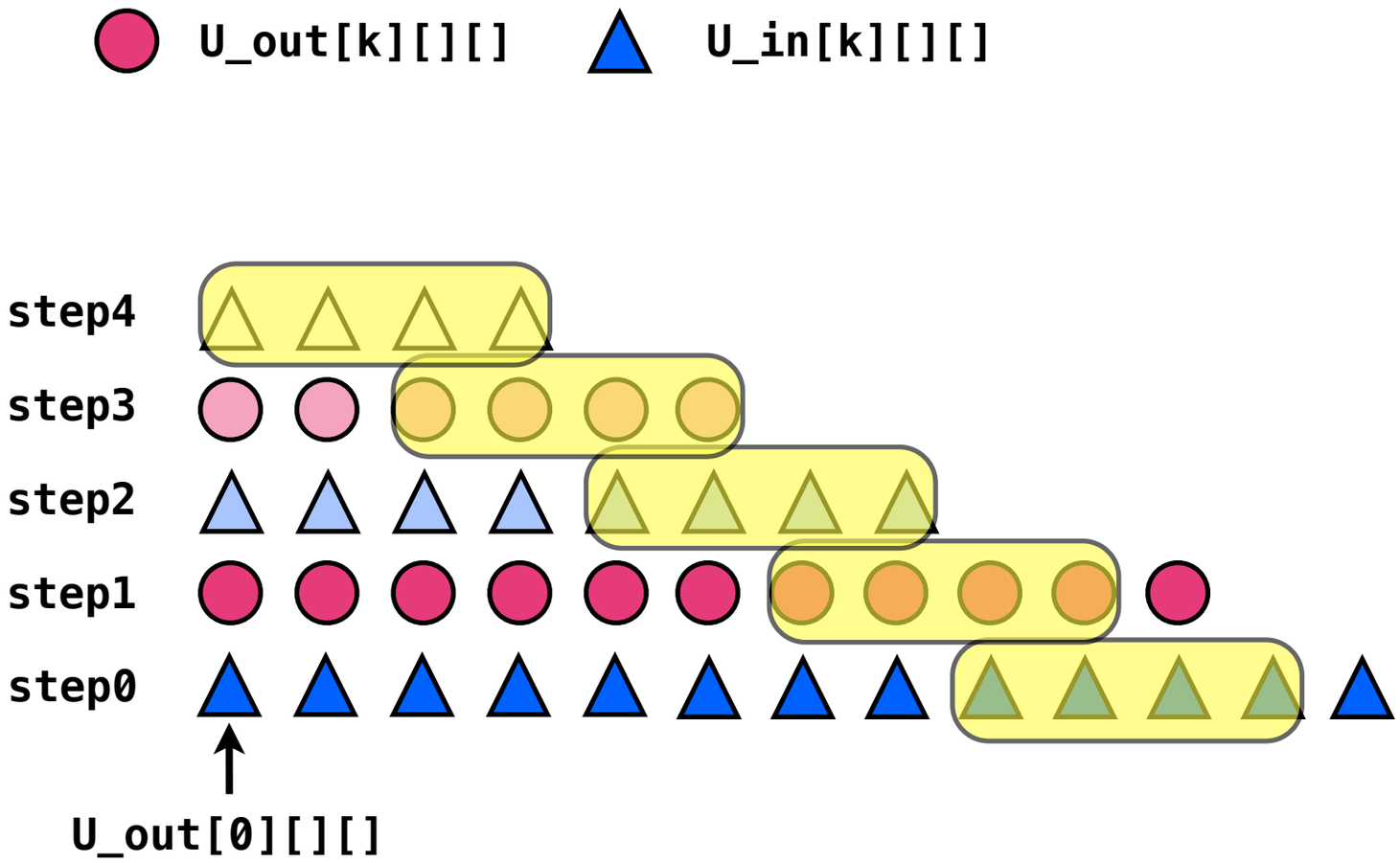
Time-Skewing Approach Description(5 steps, tile size = 4)



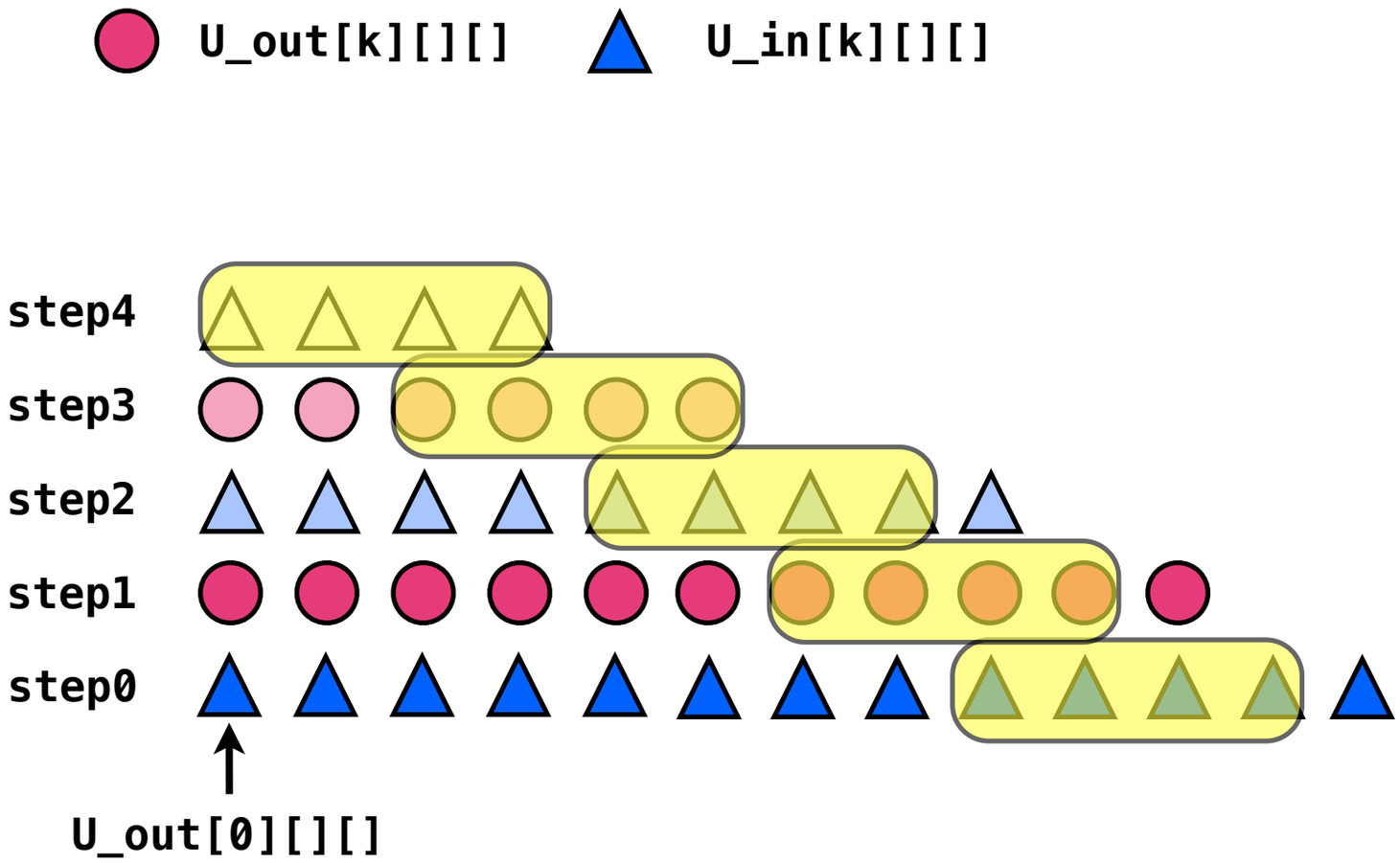
Time-Skewing Approach Description(5 steps, tile size = 4)



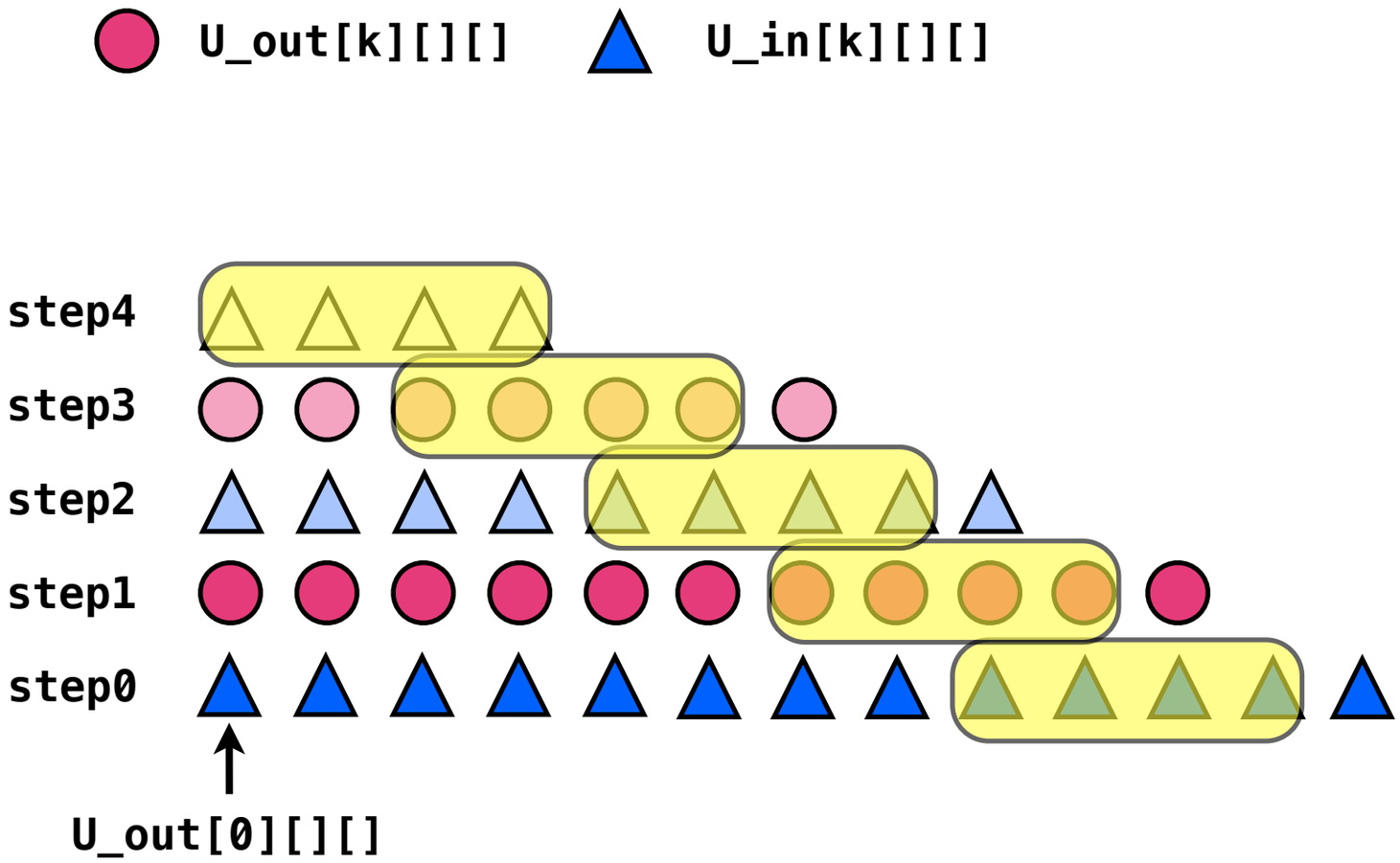
Time-Skewing Approach Description(5 steps, tile size = 4)



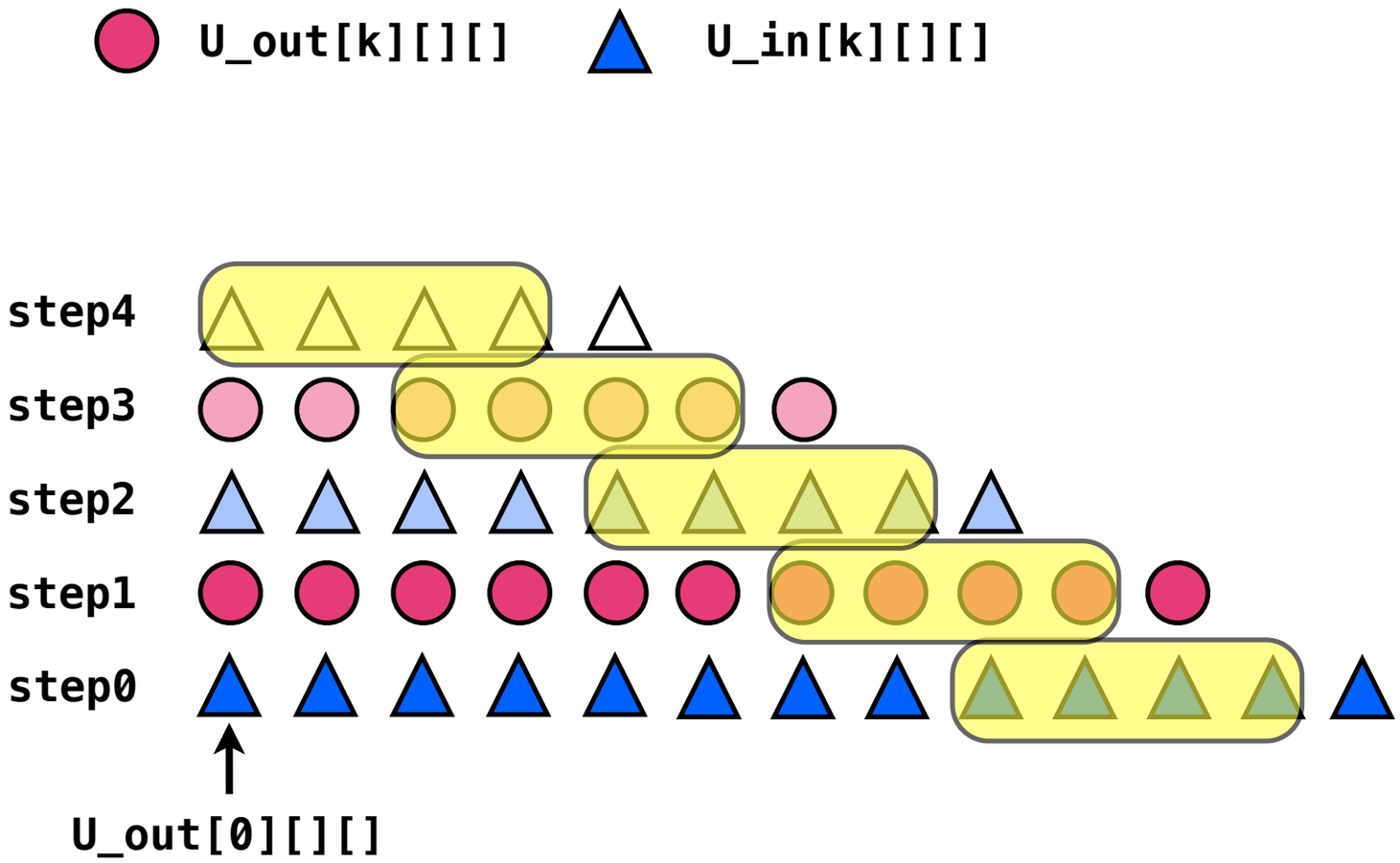
Time-Skewing Approach Description(5 steps, tile size = 4)



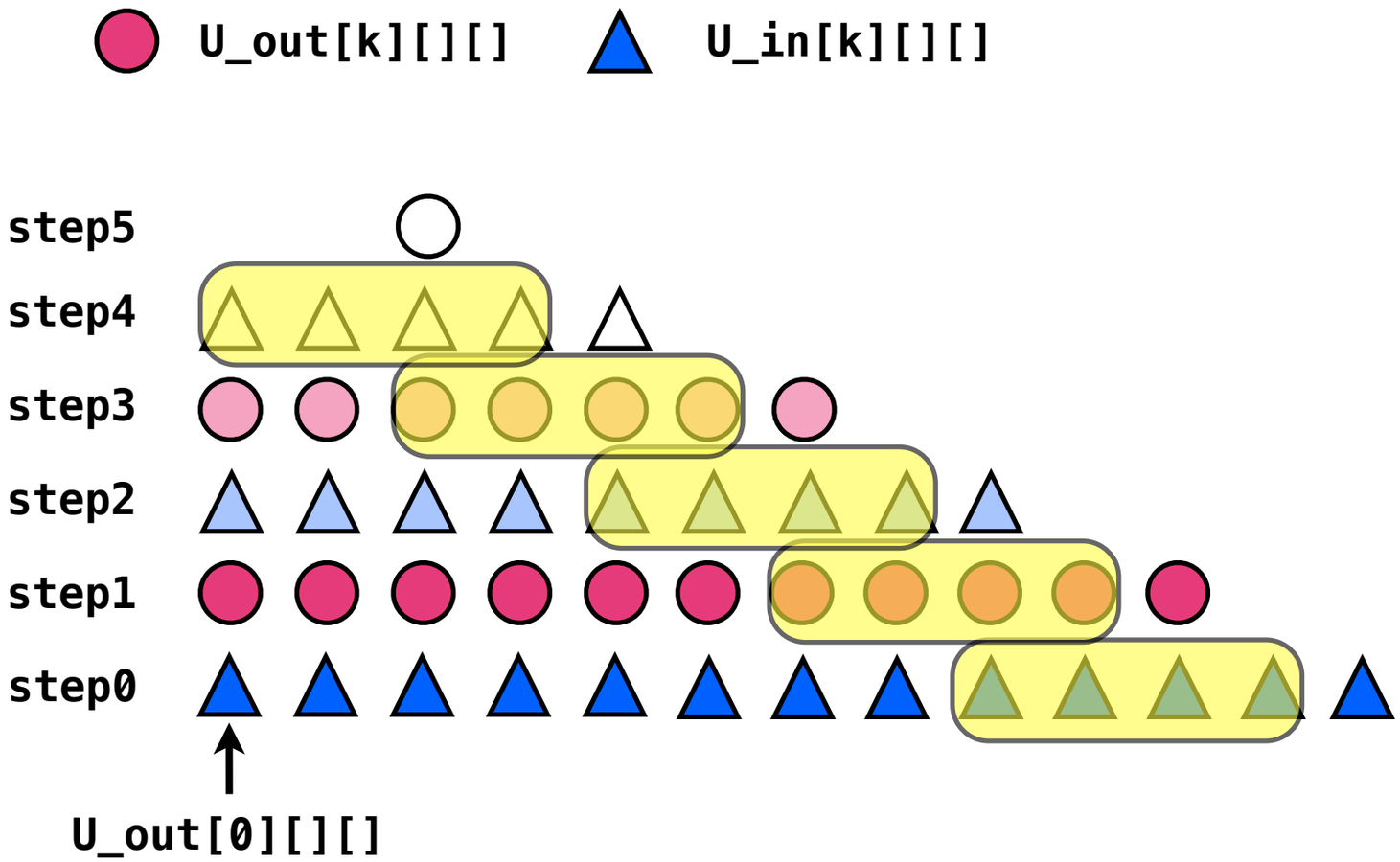
Time-Skewing Approach Description(5 steps, tile size = 4)



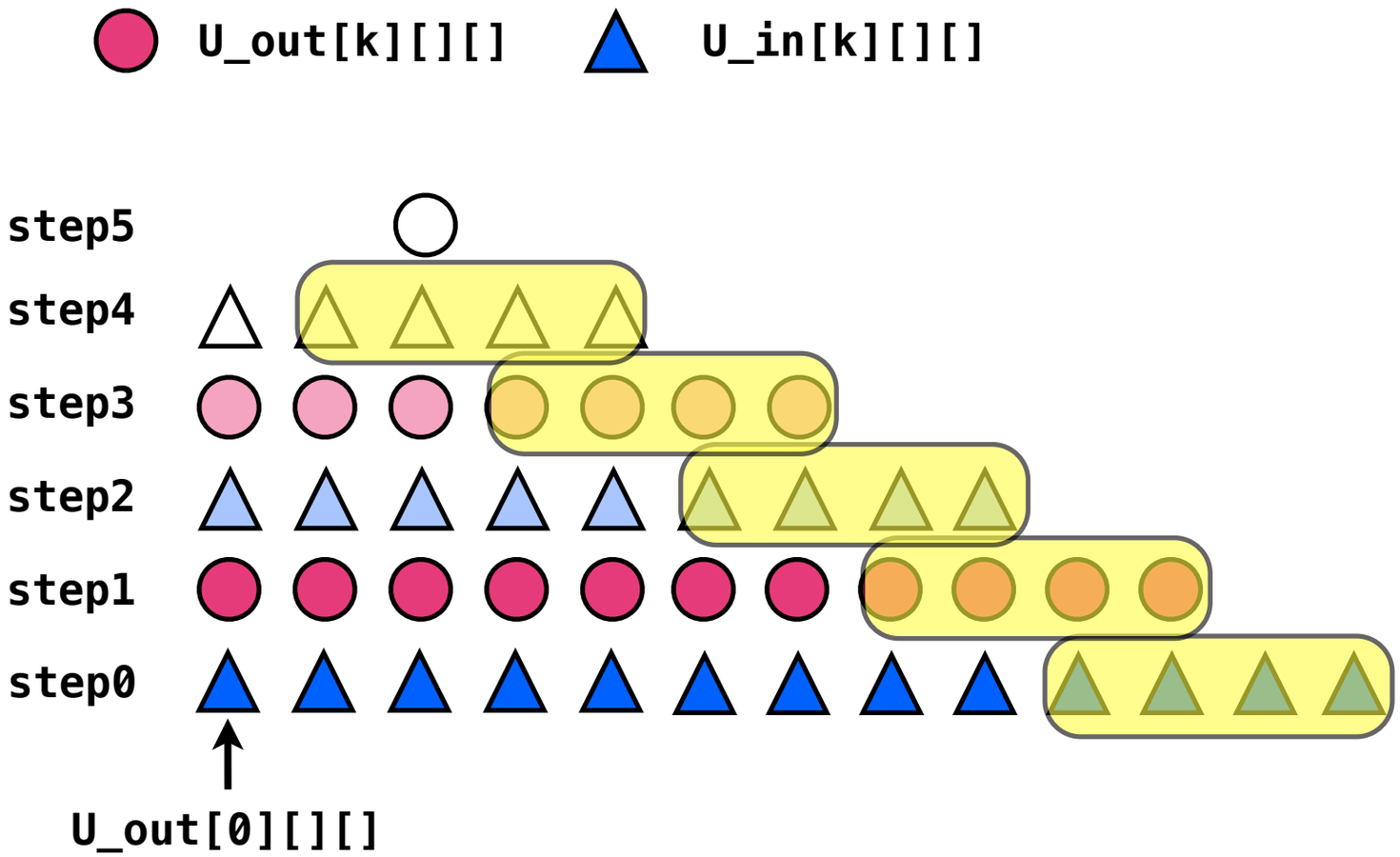
Time-Skewing Approach Description(5 steps, tile size = 4)



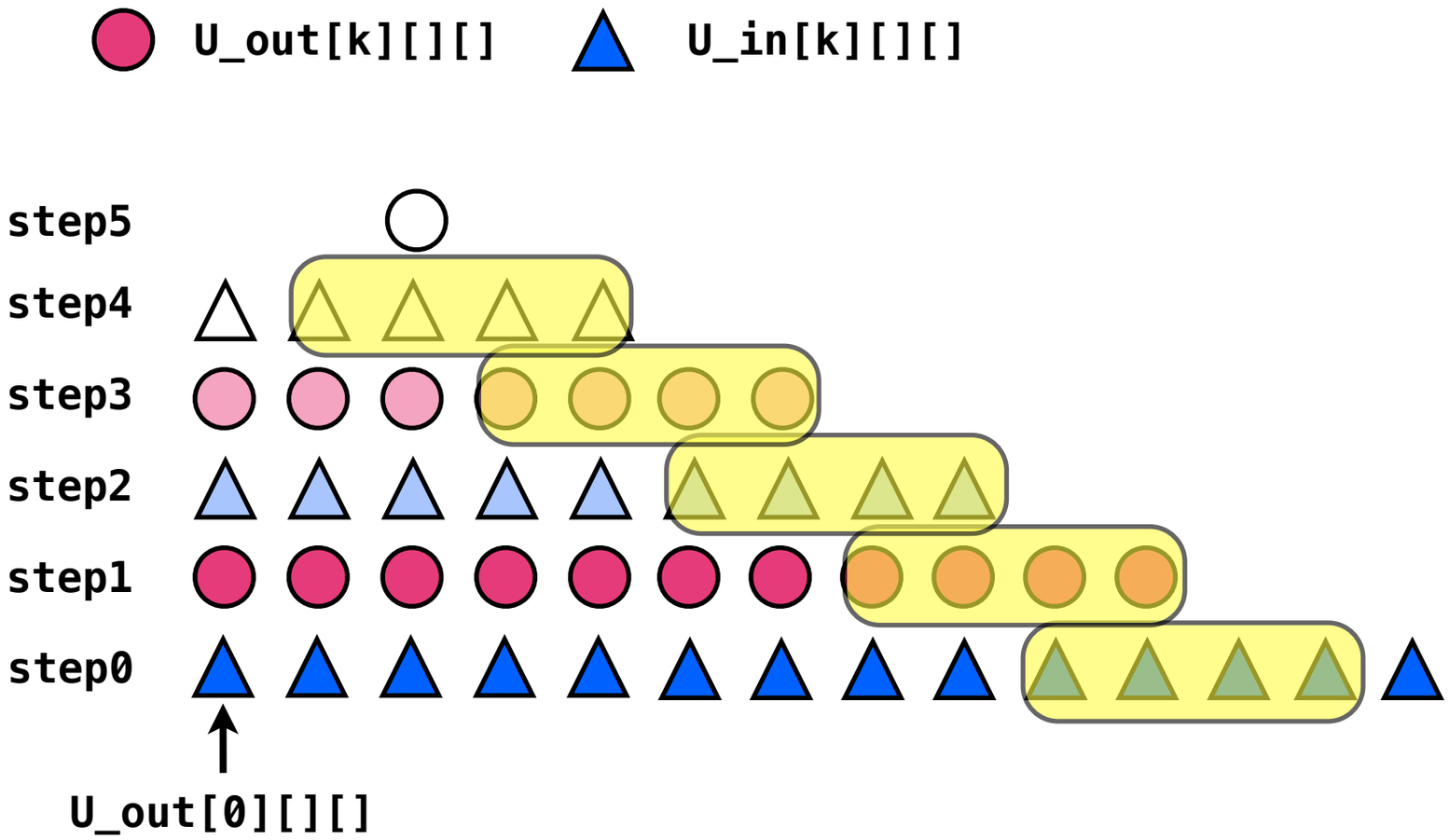
Time-Skewing Approach Description(5 steps, tile size = 4)



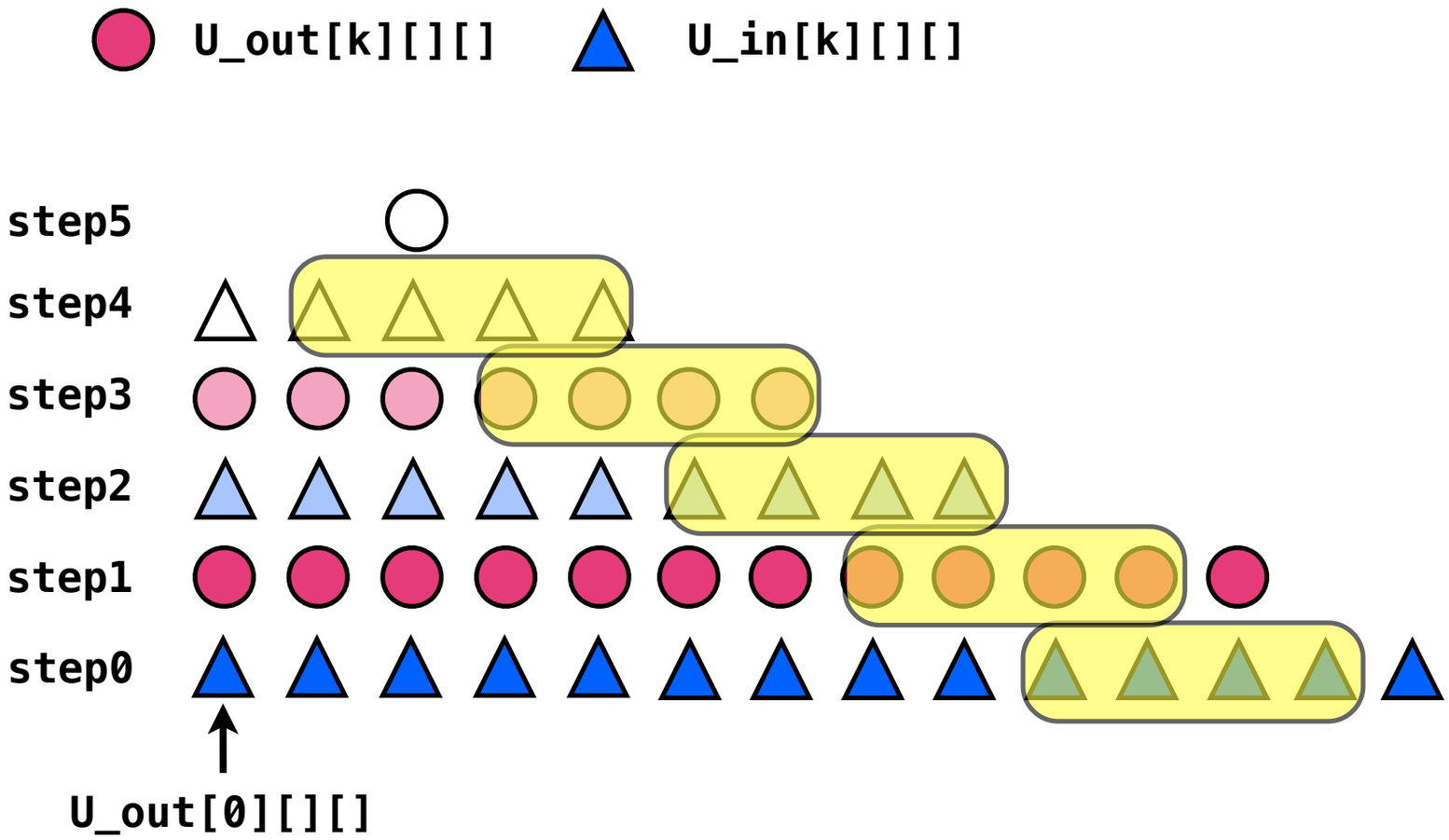
Time-Skewing Approach Description(5 steps, tile size = 4)



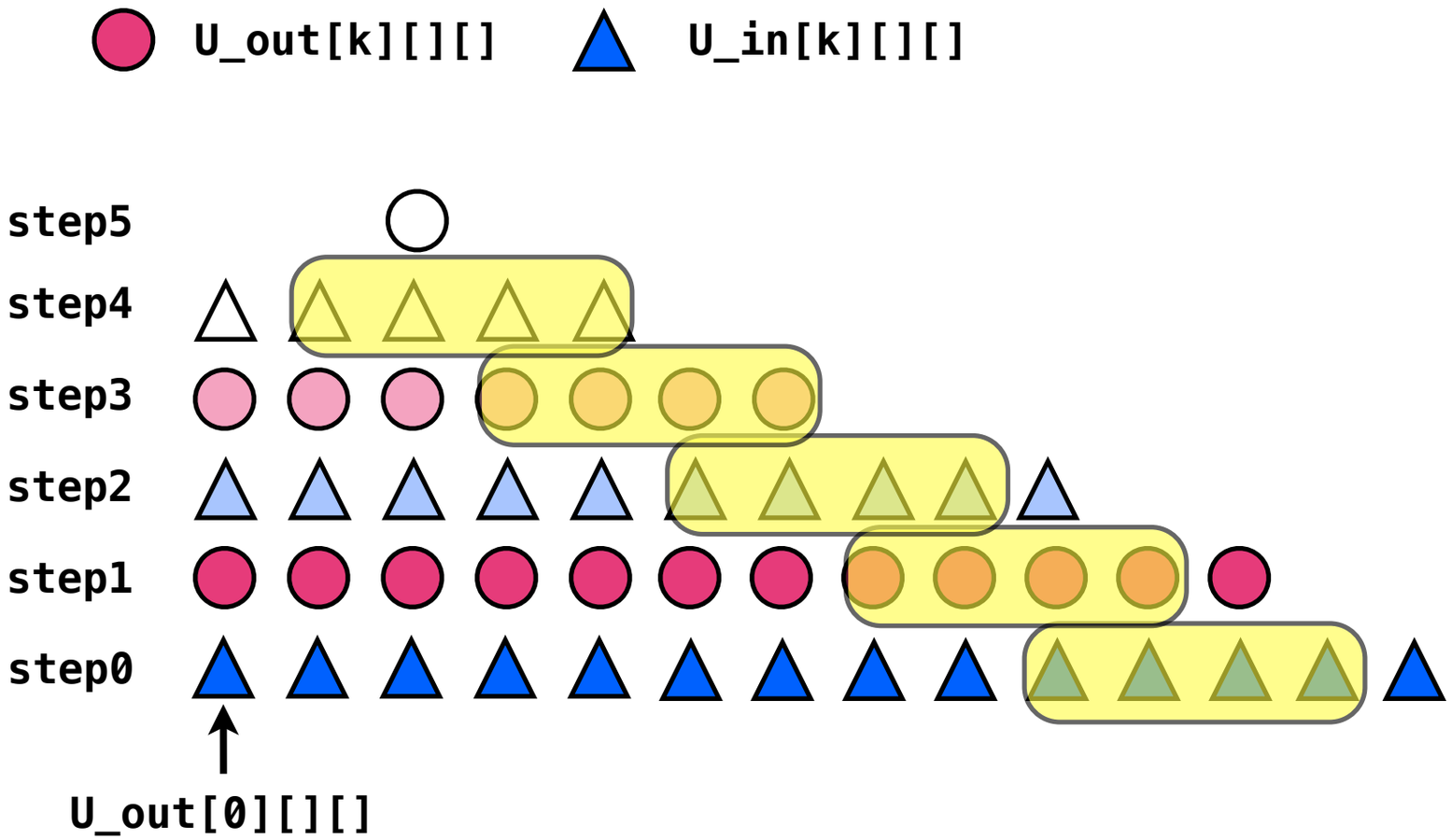
Time-Skewing Approach Description(5 steps, tile size = 4)



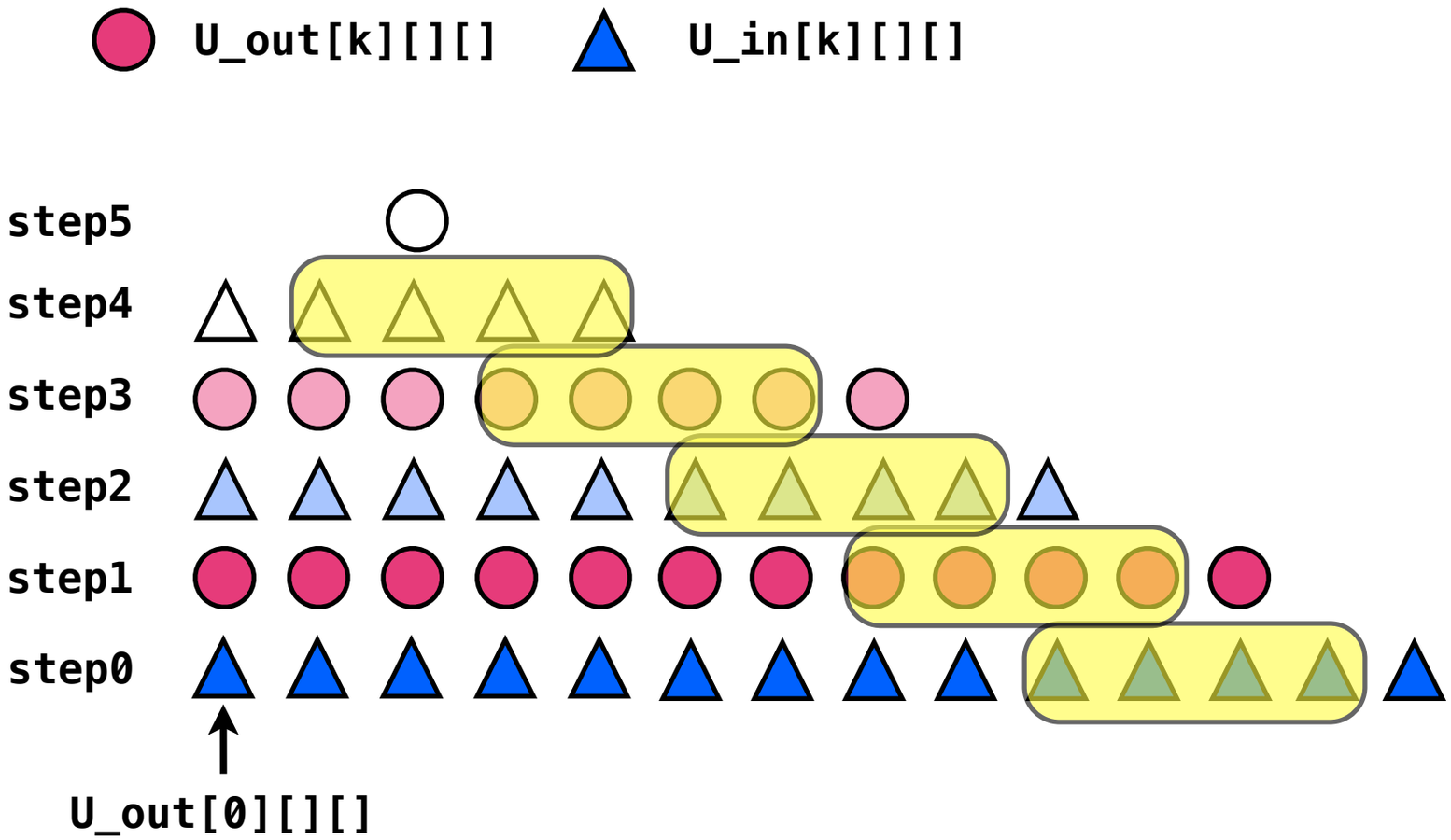
Time-Skewing Approach Description(5 steps, tile size = 4)



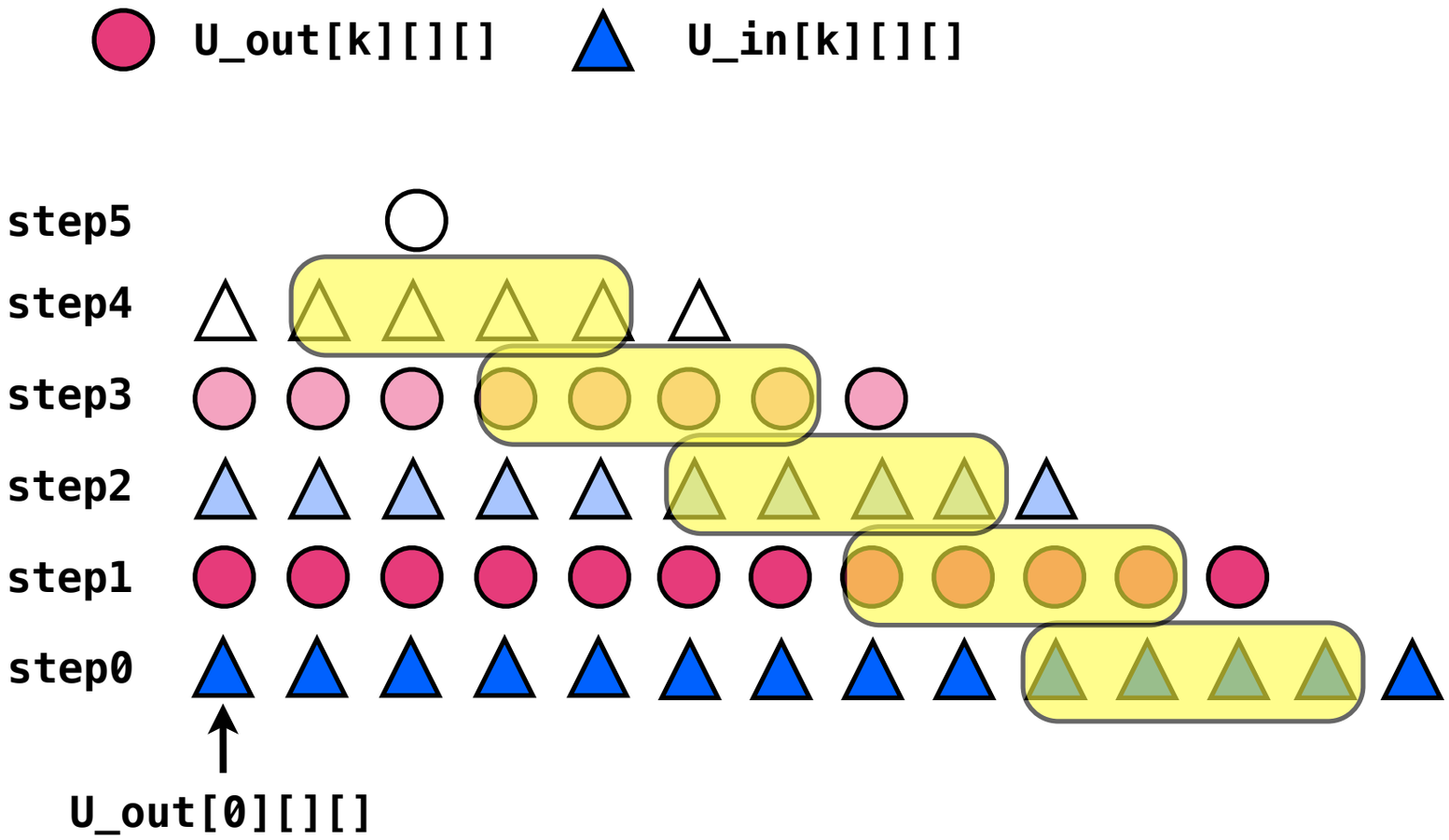
Time-Skewing Approach Description(5 steps, tile size = 4)



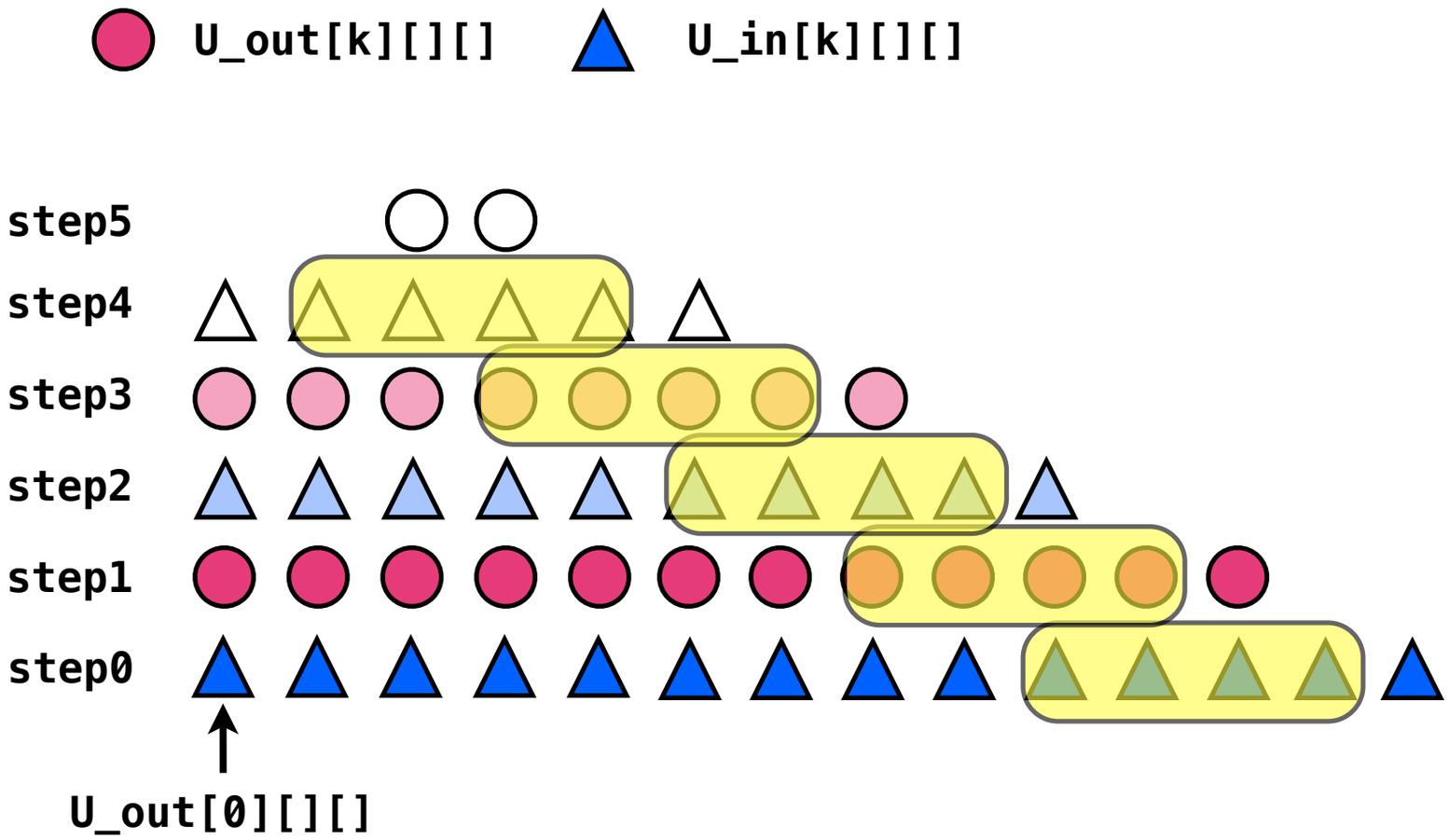
Time-Skewing Approach Description(5 steps, tile size = 4)



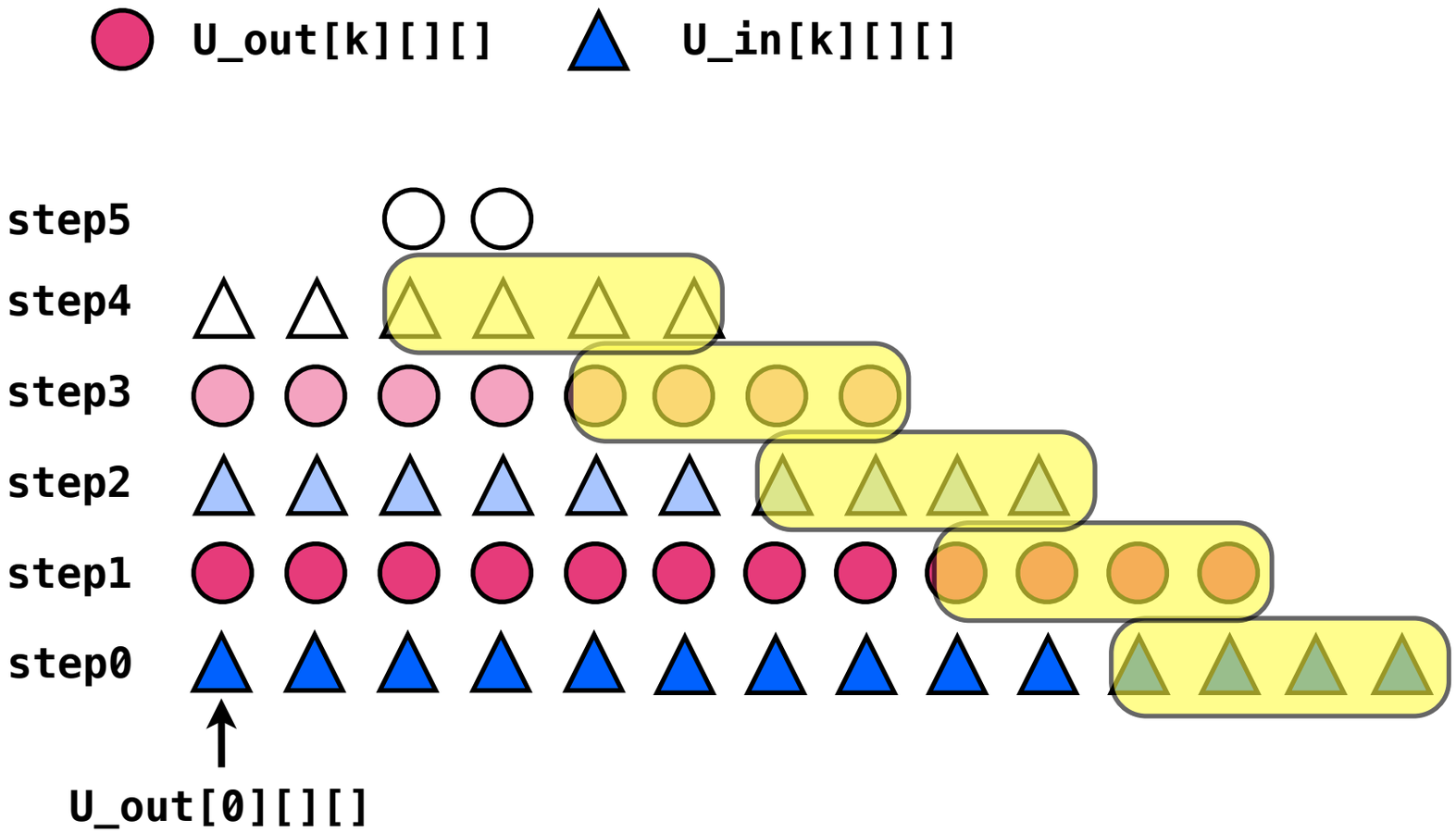
Time-Skewing Approach Description(5 steps, tile size = 4)



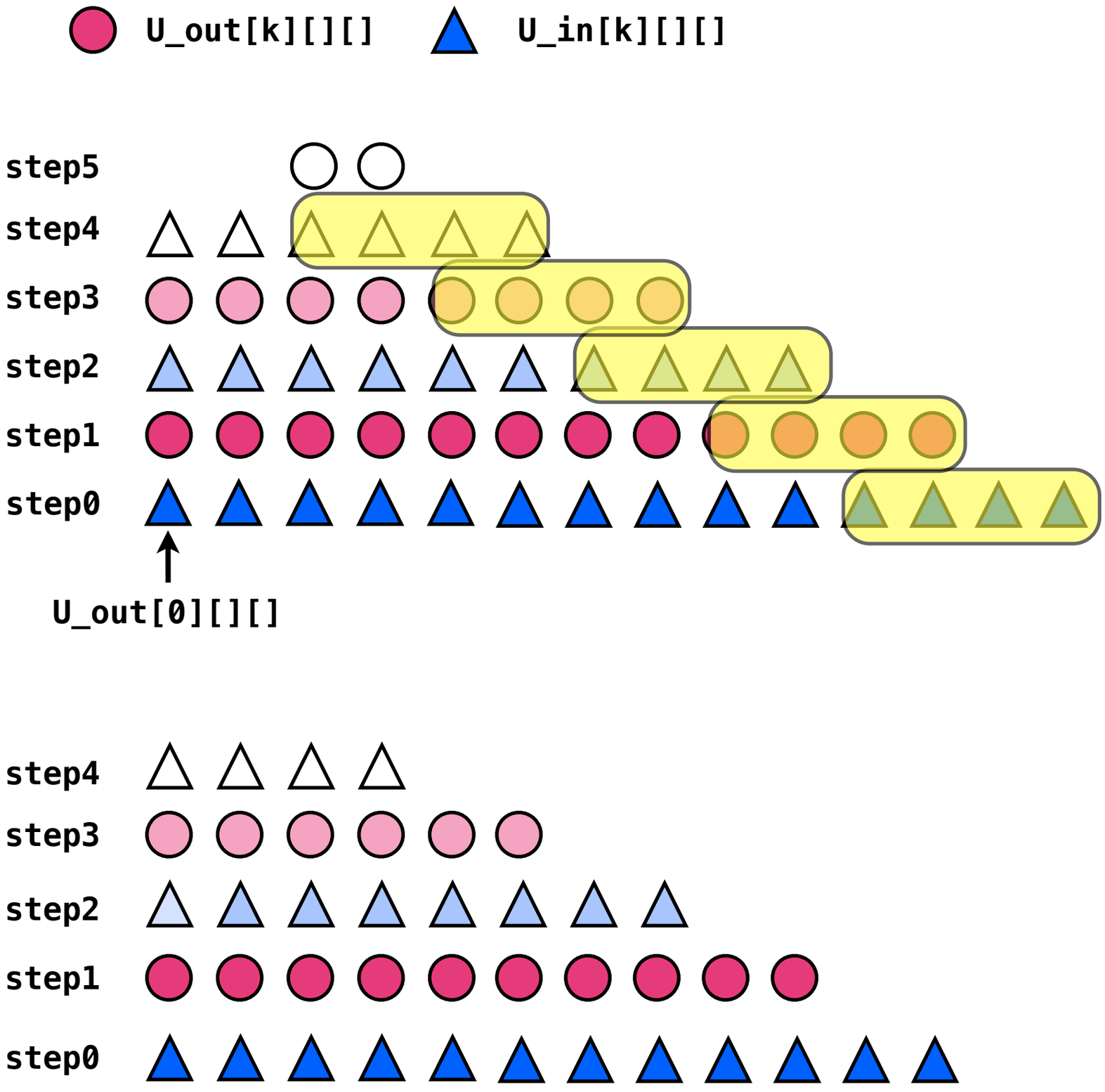
Time-Skewing Approach Description(5 steps, tile size = 4)



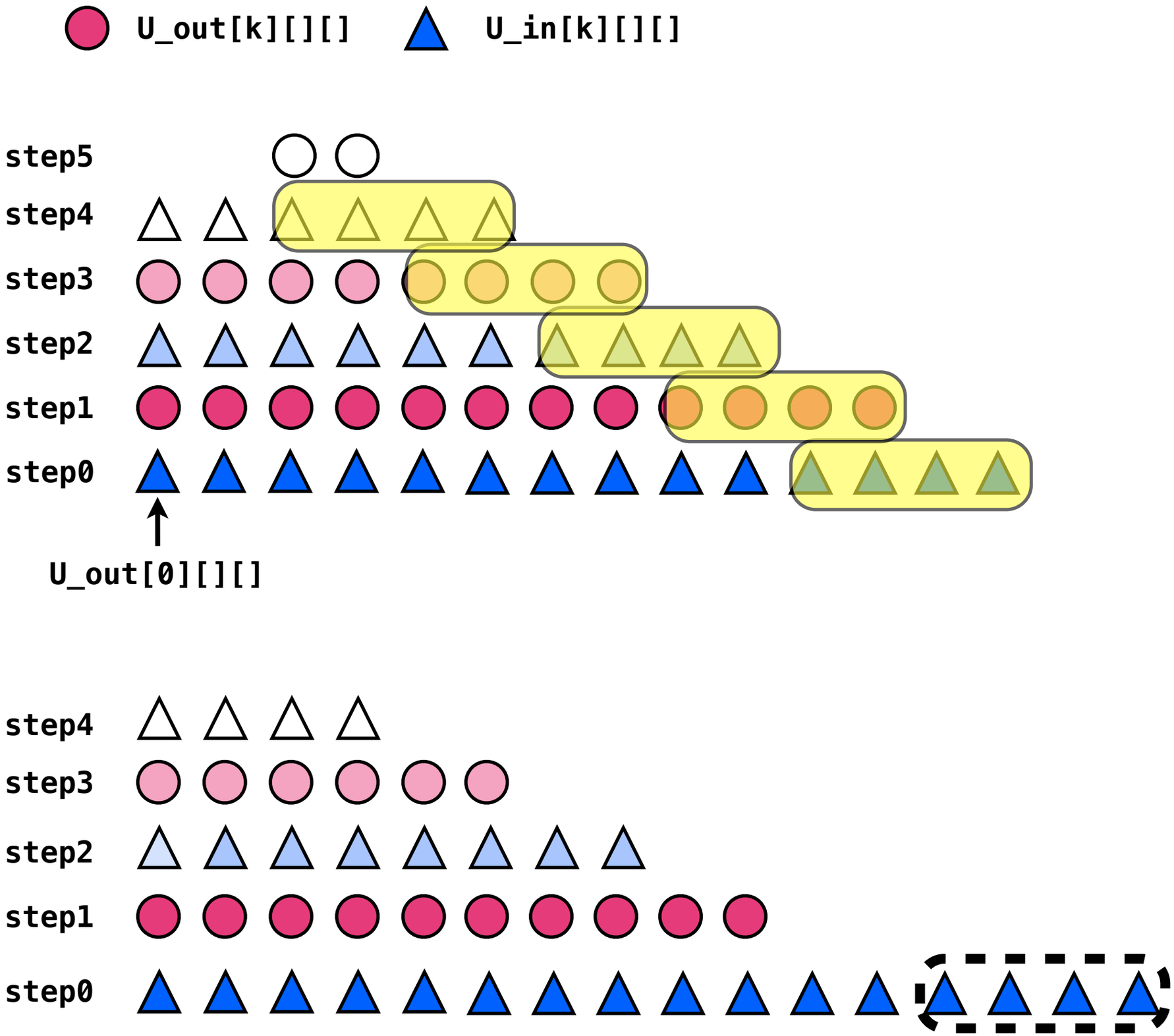
Time-Skewing Approach Description(5 steps, tile size = 4)



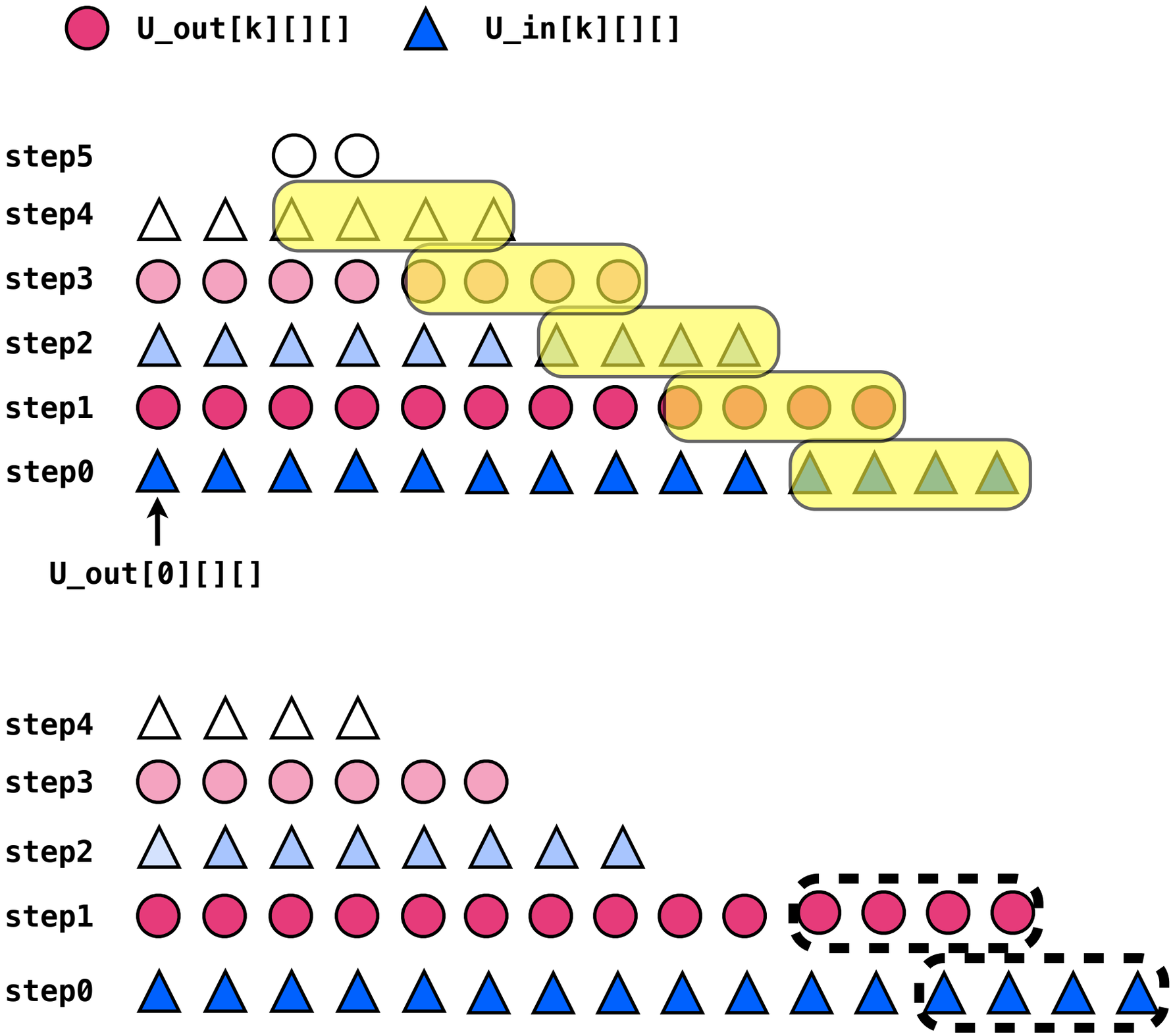
Time-Skewing Approach Description(5 steps, tile size = 4)



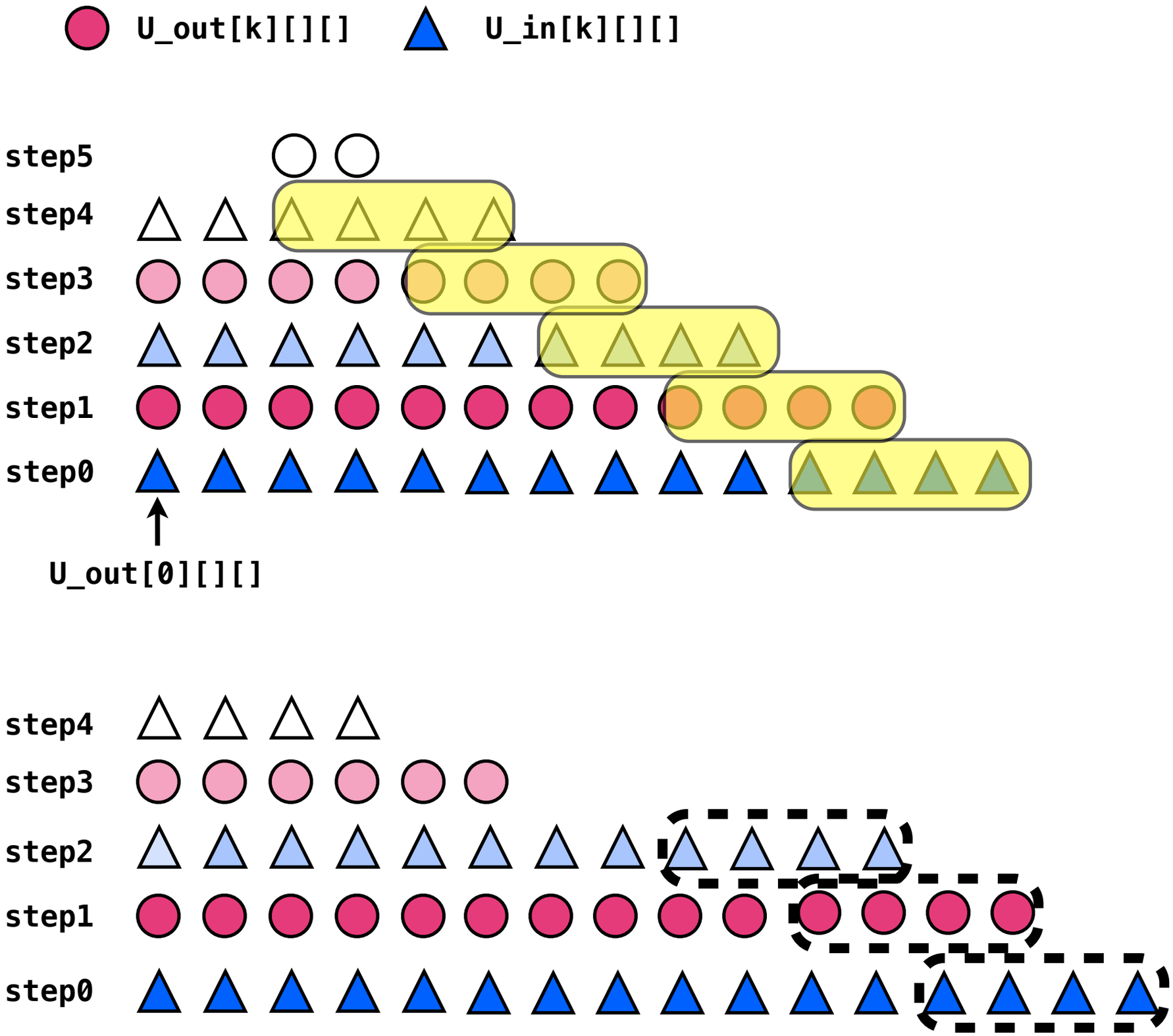
Time-Skewing Approach Description(5 steps, tile size = 4)



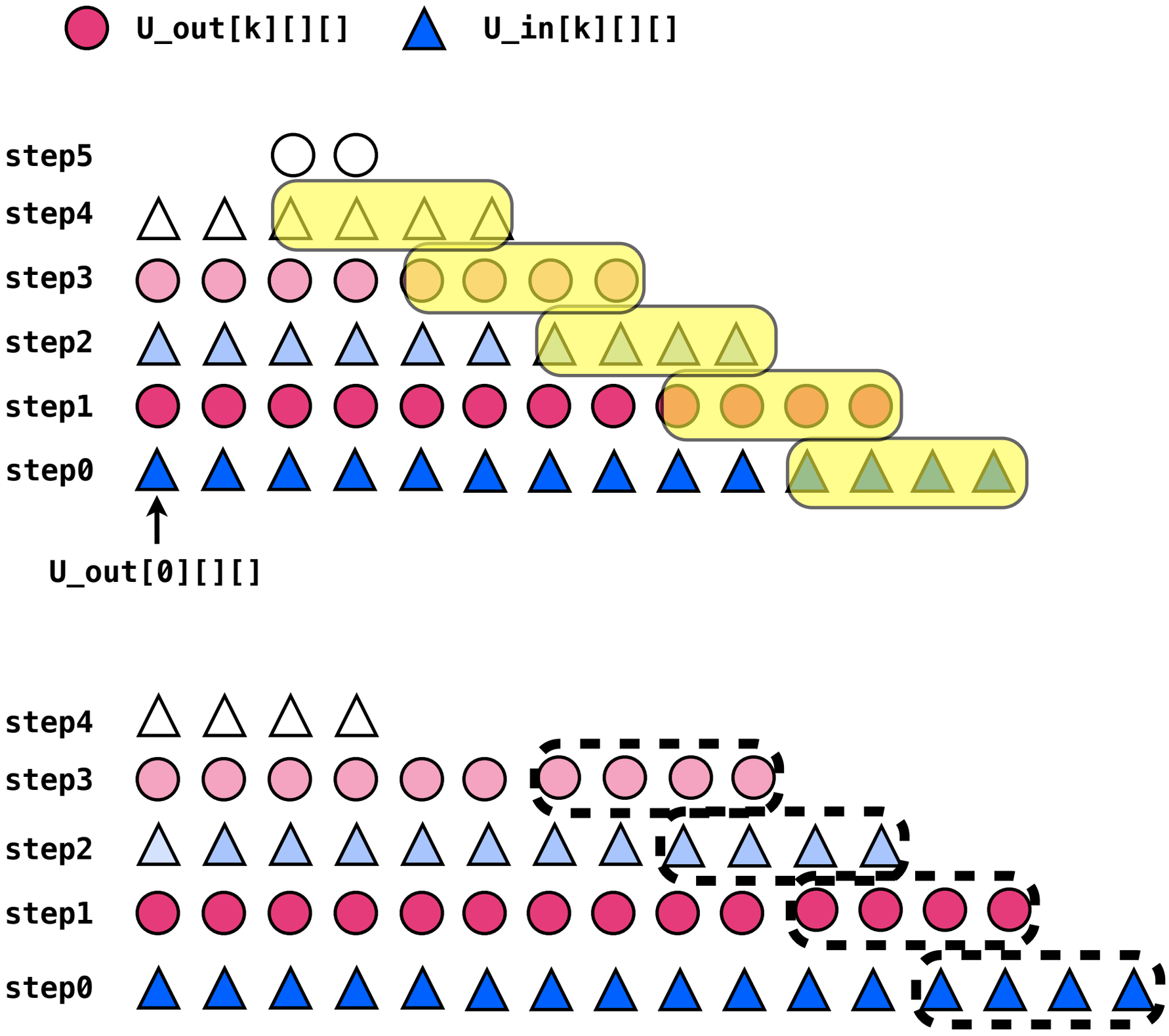
Time-Skewing Approach Description(5 steps, tile size = 4)



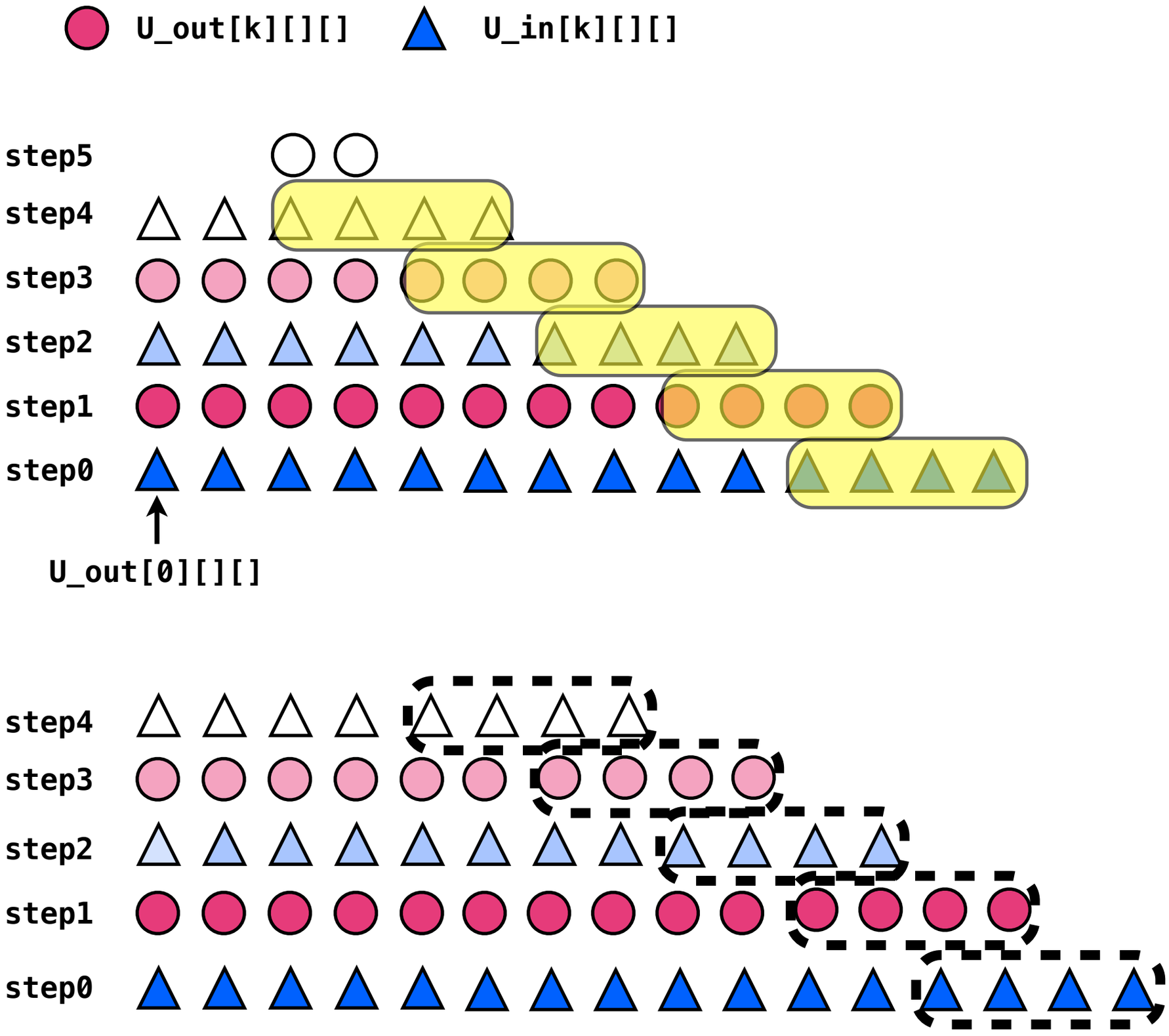
Time-Skewing Approach Description(5 steps, tile size = 4)



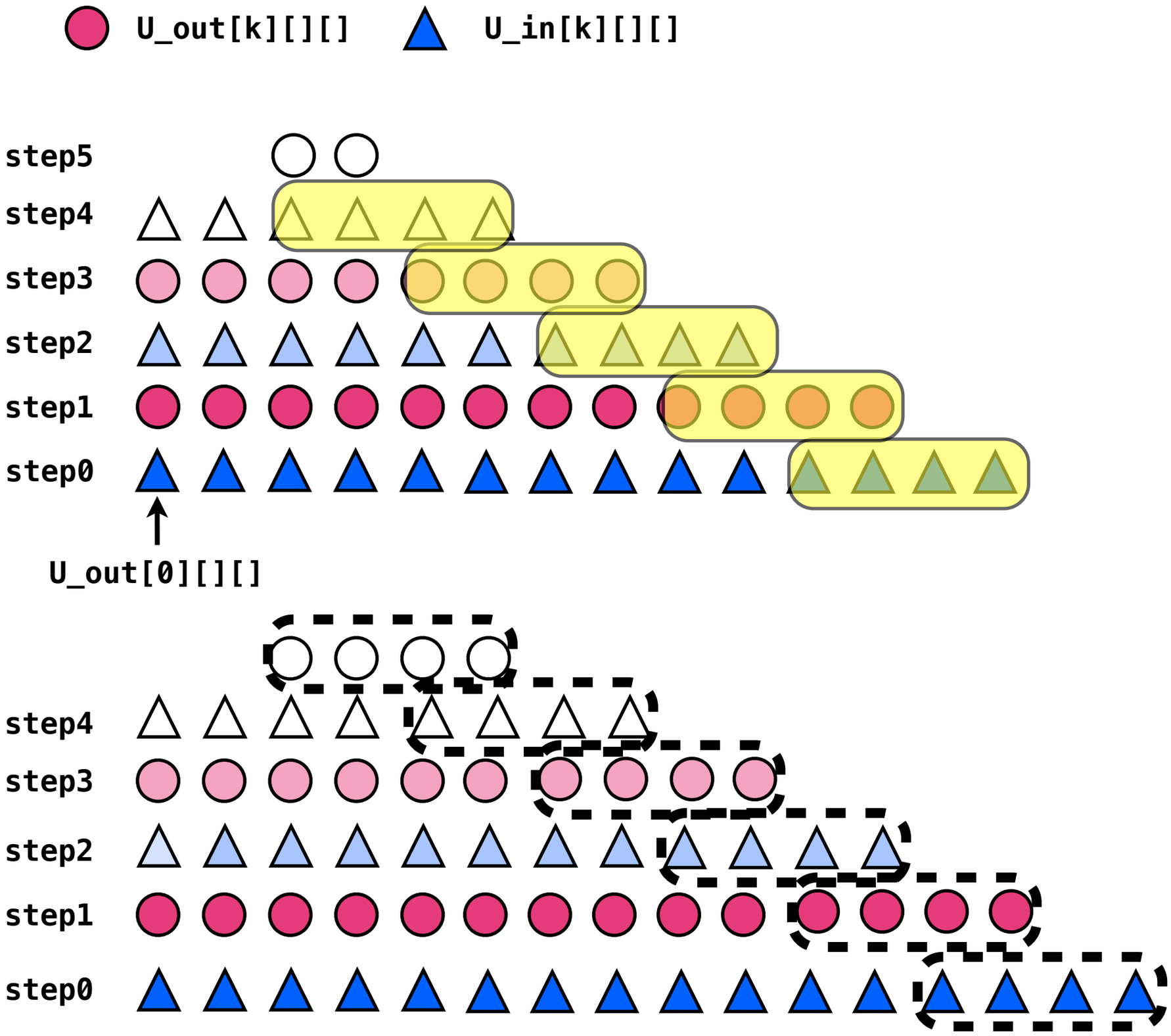
Time-Skewing Approach Description(5 steps, tile size = 4)



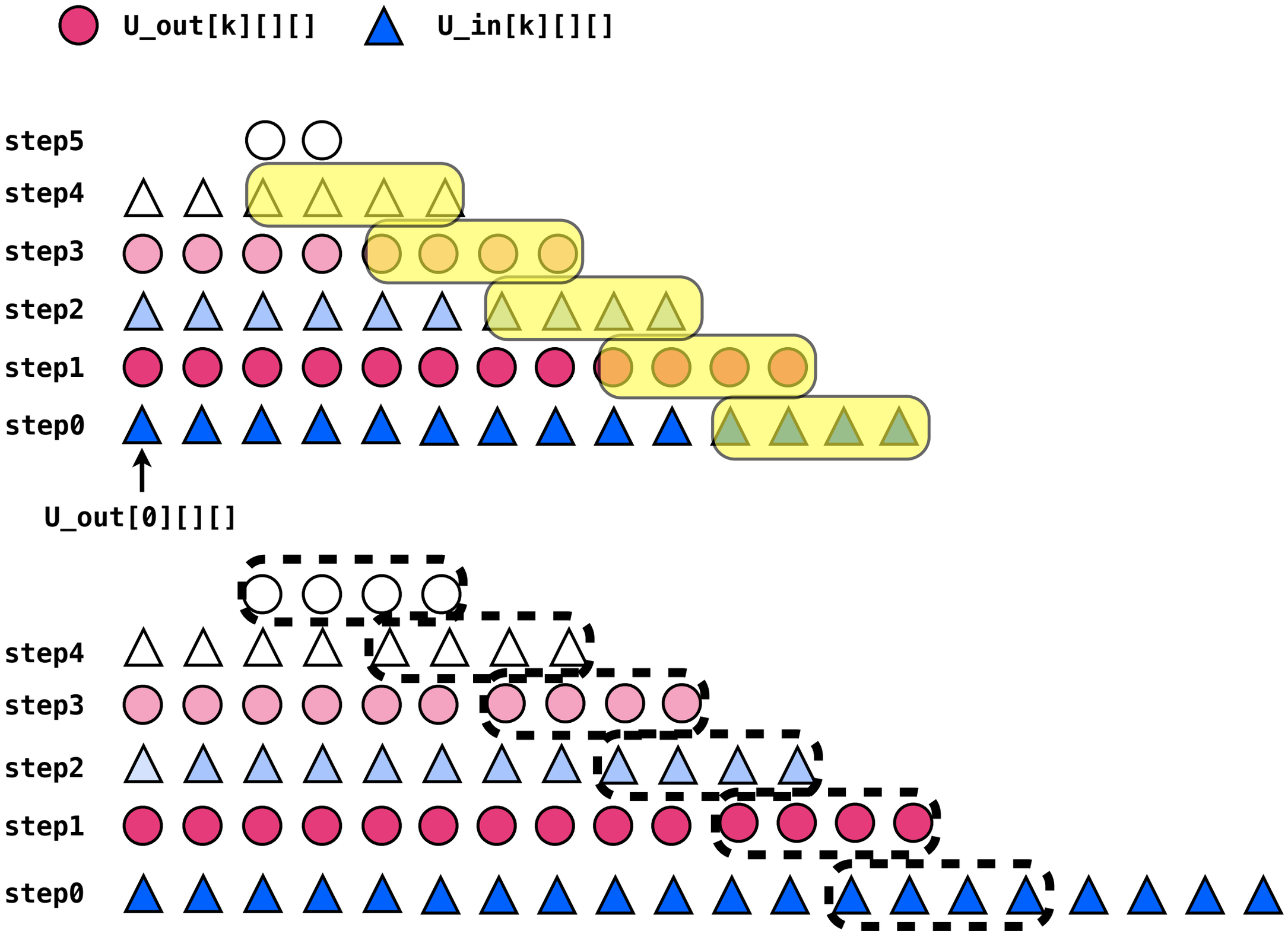
Time-Skewing Approach Description(5 steps, tile size = 4)



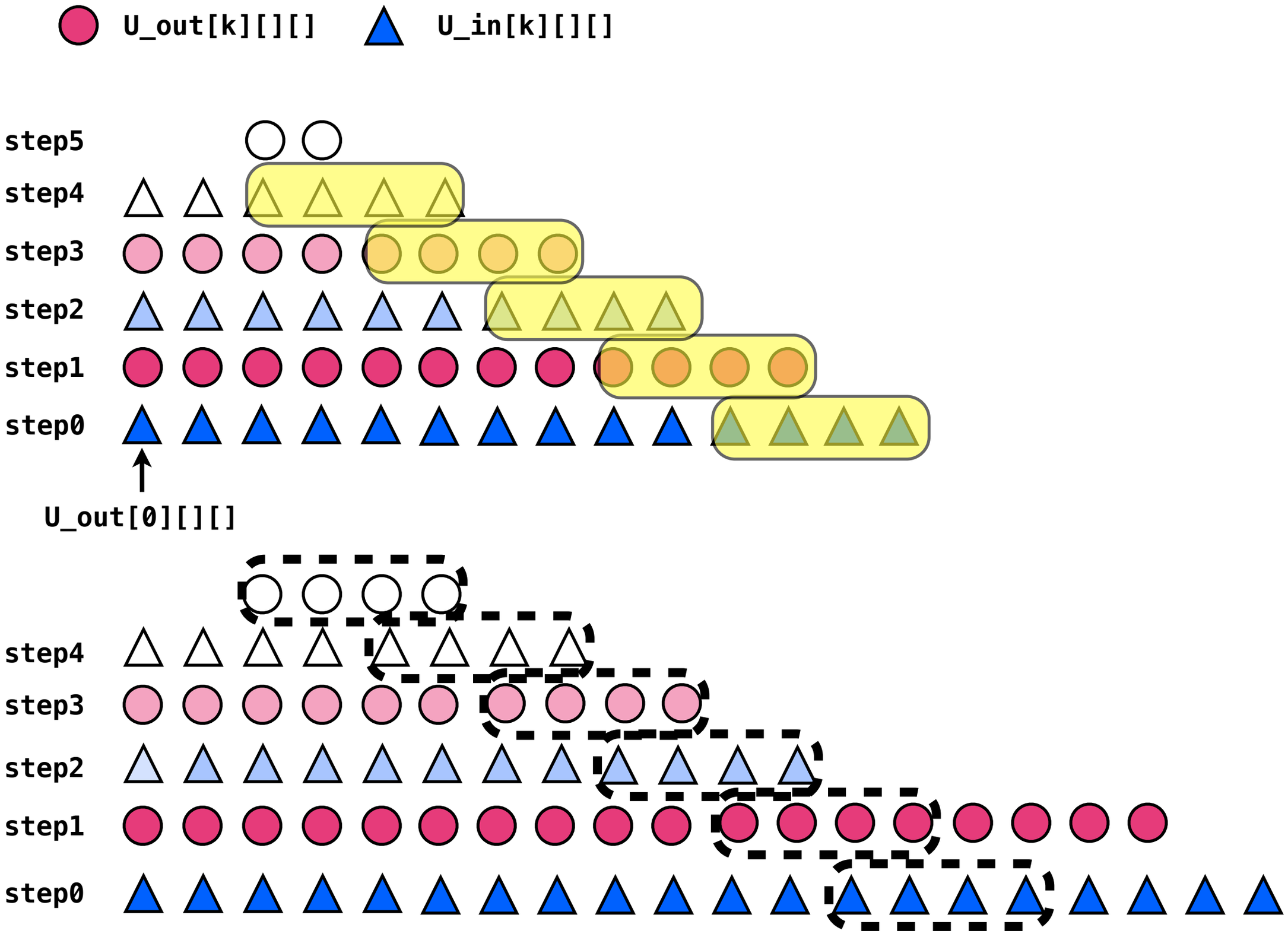
Time-Skewing Approach Description(5 steps, tile size = 4)



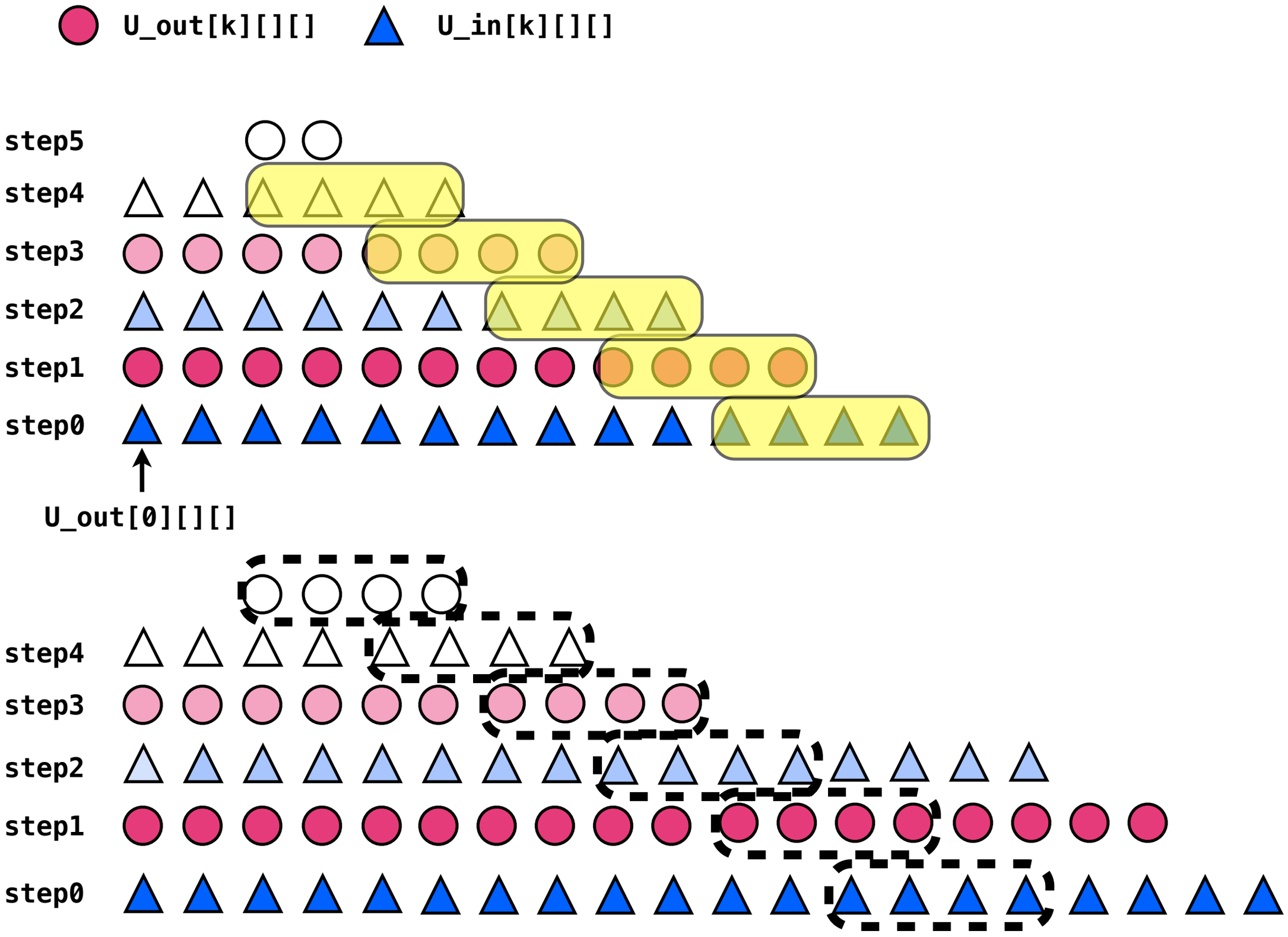
Time-Skewing Approach Description(5 steps, tile size = 4)



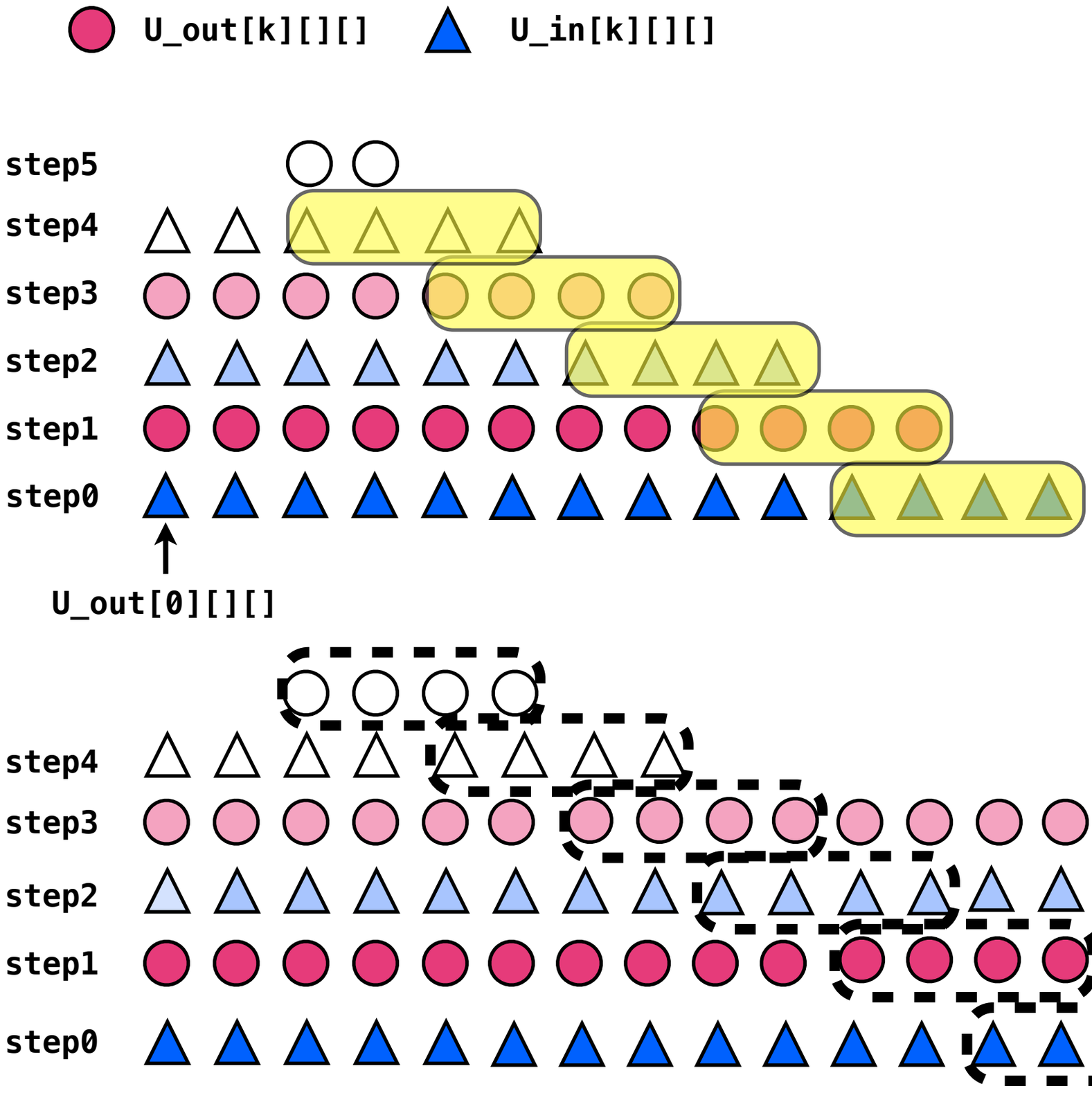
Time-Skewing Approach Description(5 steps, tile size = 4)



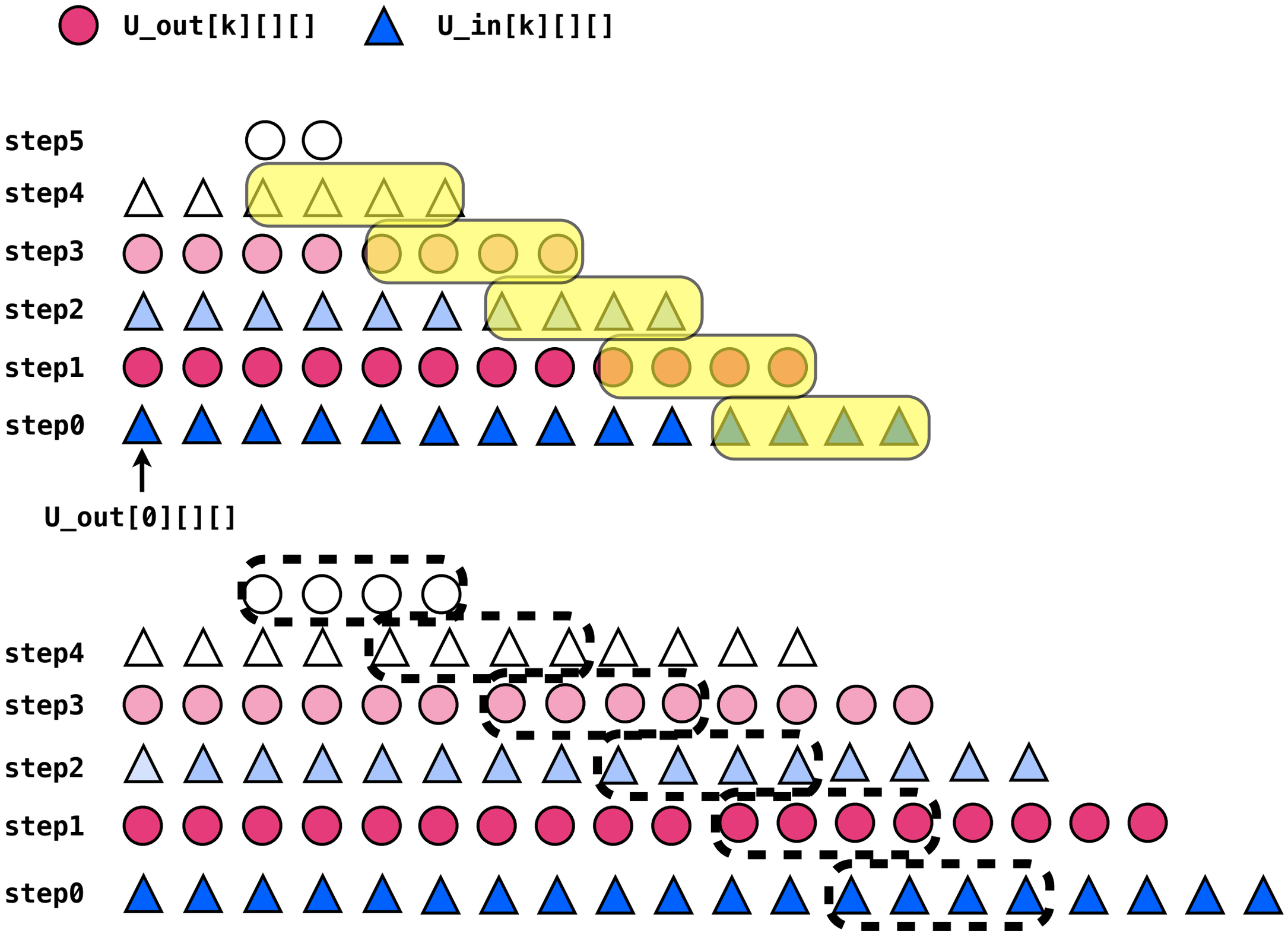
Time-Skewing Approach Description(5 steps, tile size = 4)



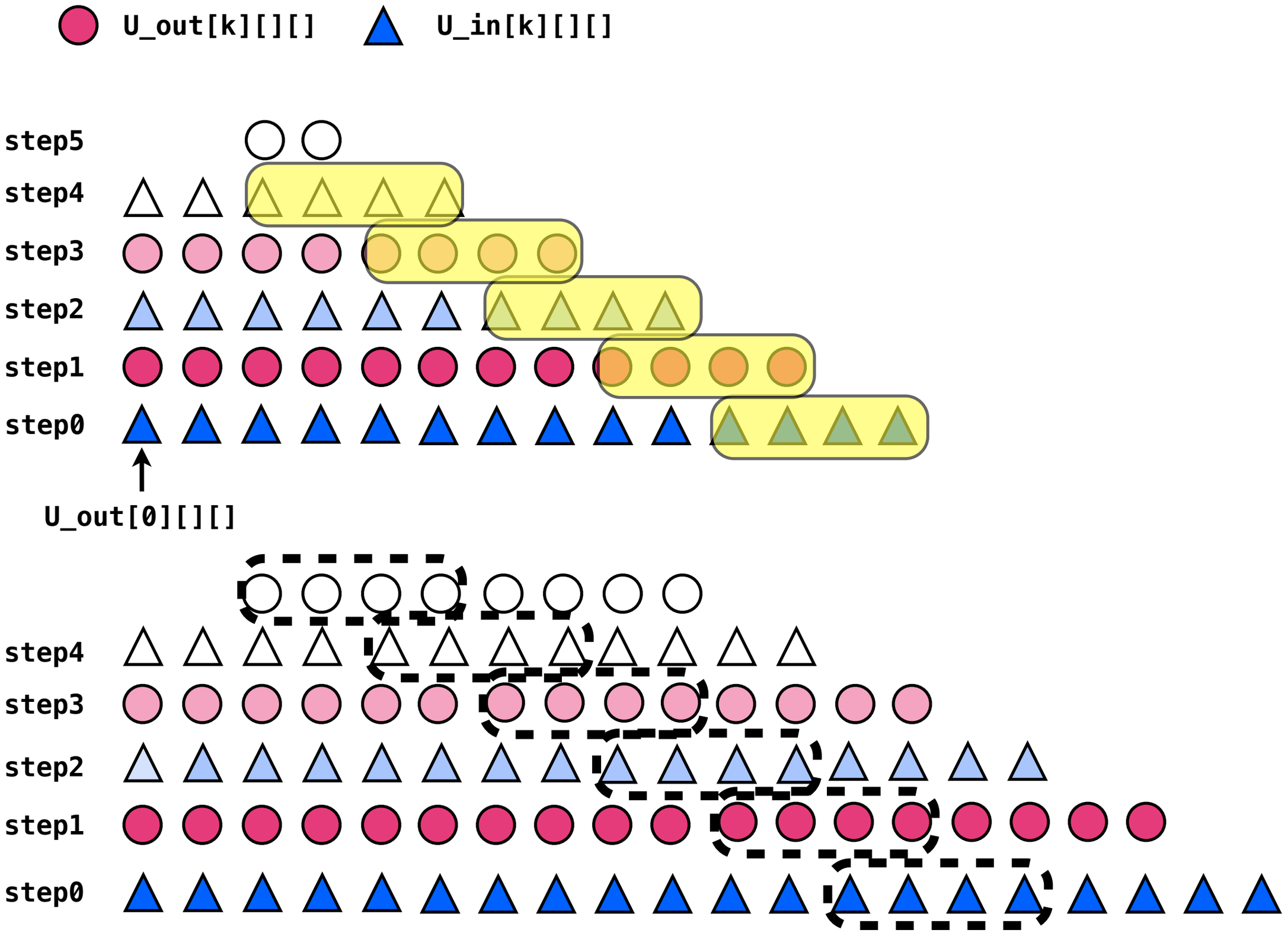
Time-Skewing Approach Description(5 steps, tile size = 4)



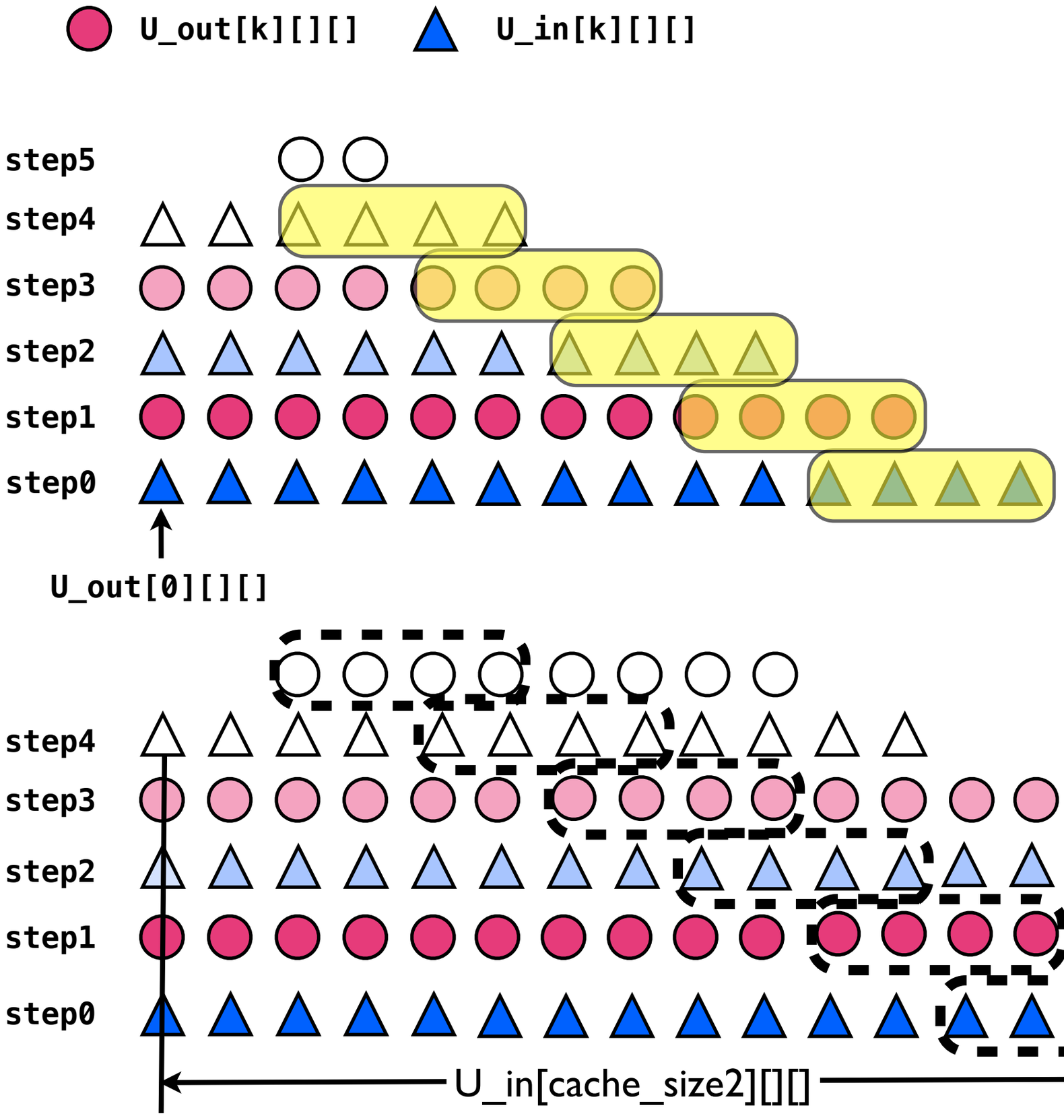
Time-Skewing Approach Description(5 steps, tile size = 4)



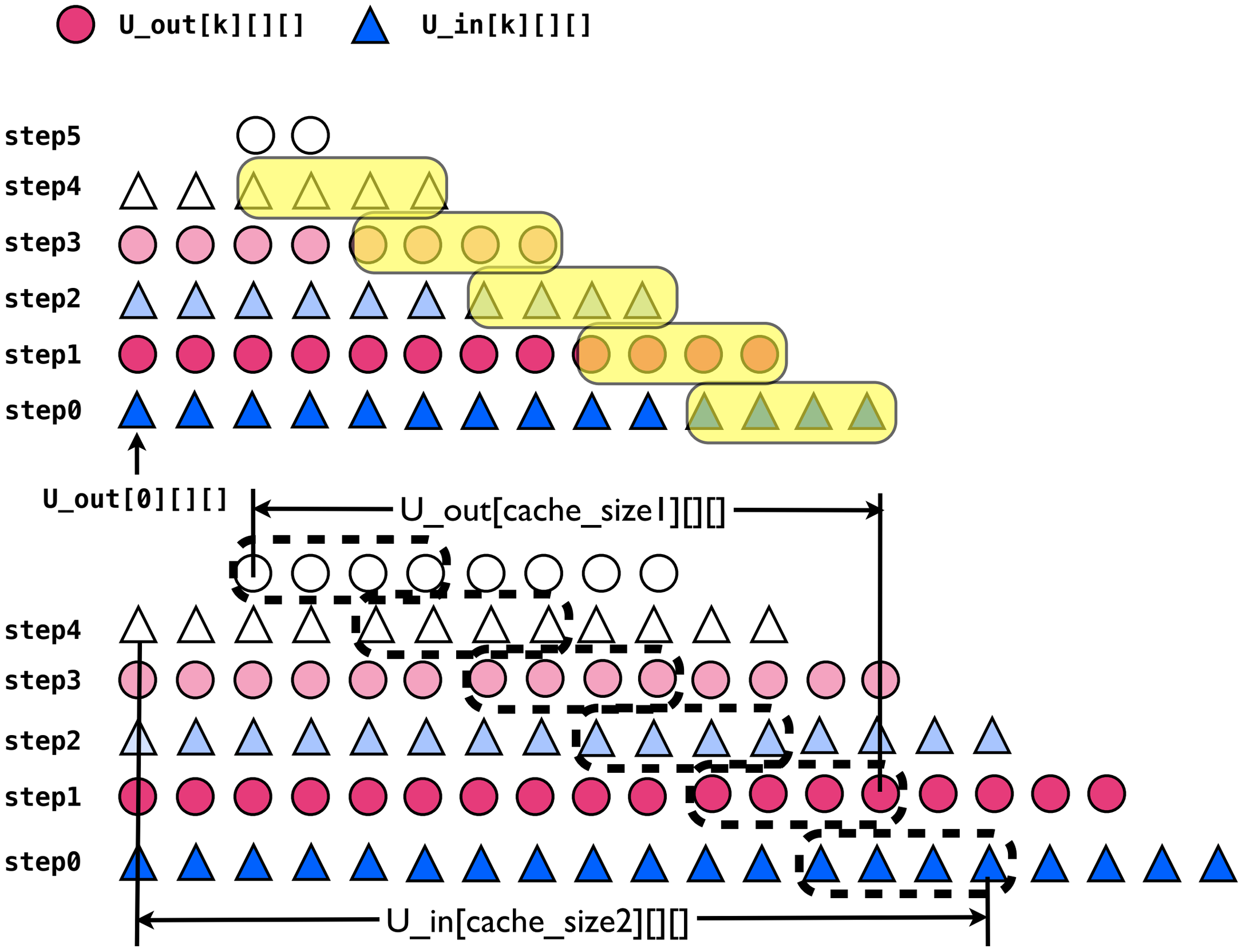
Time-Skewing Approach Description(5 steps, tile size = 4)



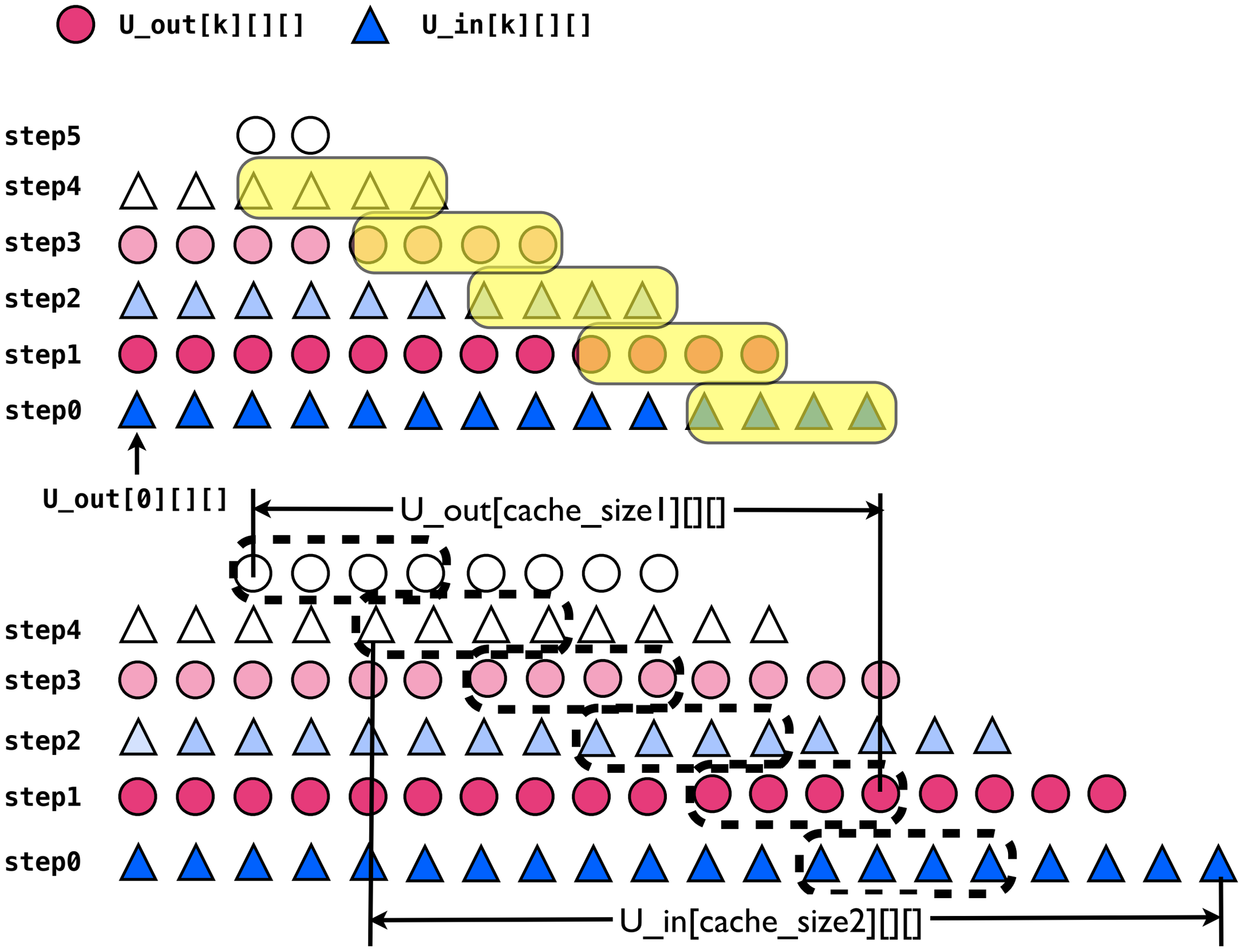
Time-Skewing Approach Description(5 steps, tile size = 4)



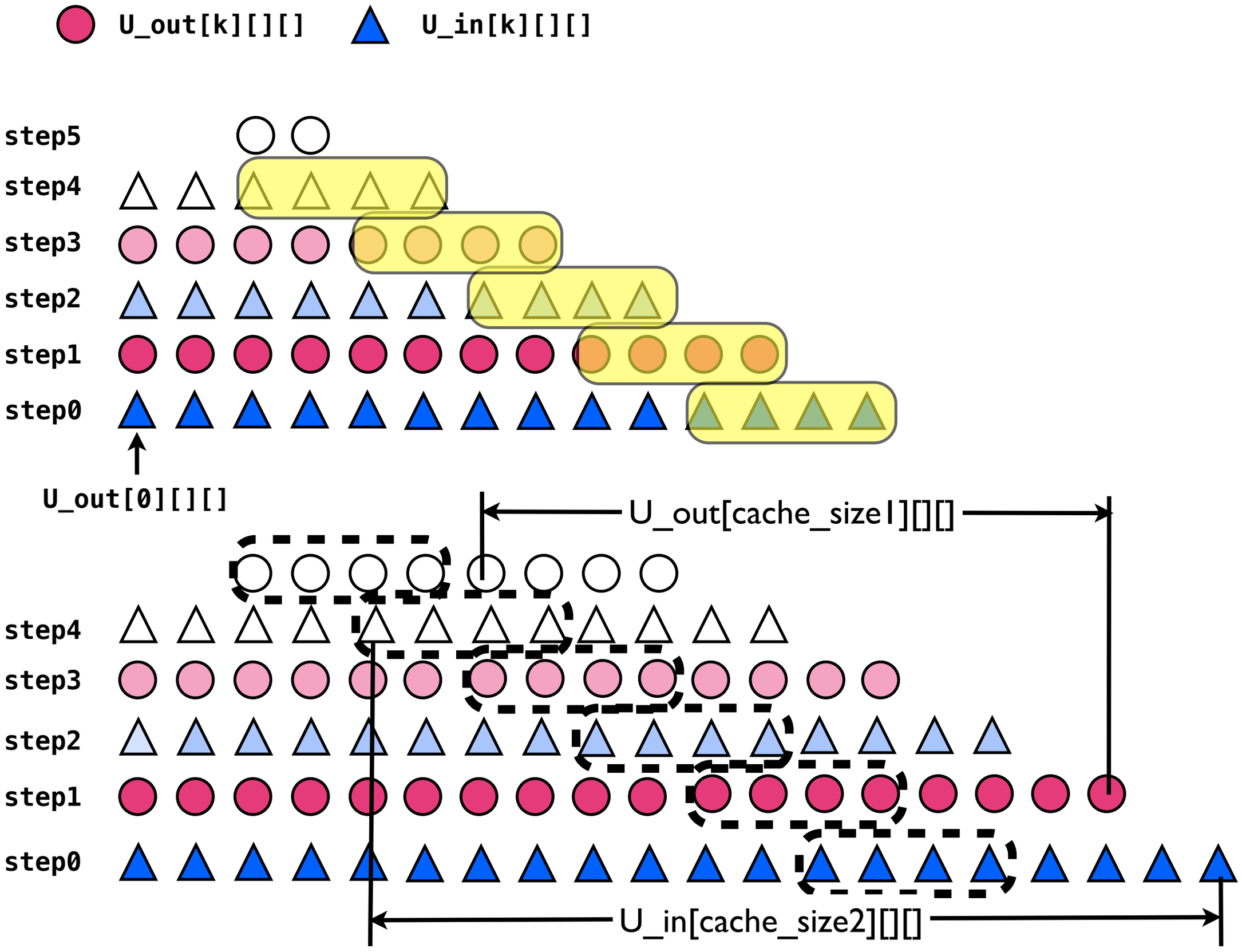
Time-Skewing Approach Description(5 steps, tile size = 4)



Time-Skewing Approach Description(5 steps, tile size = 4)



Time-Skewing Approach Description(5 steps, tile size = 4)



Time-Skewing Approach Analysis

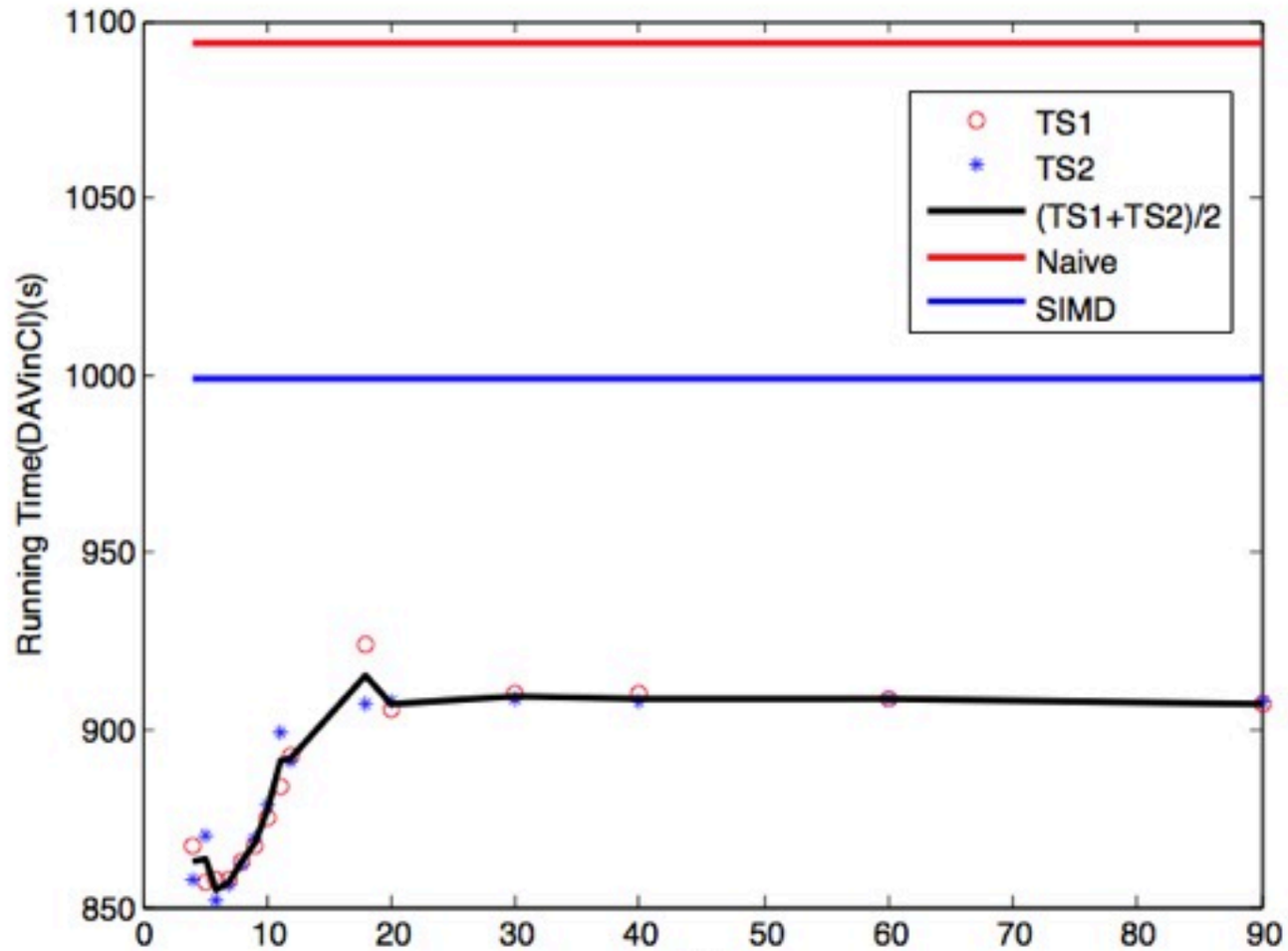
TS denotes the number of time steps performed in one iteration described in the previous slide
The number of planes expected in cache (cache_size1 + cache_size2): 4*TS + 8
(4*TS + 8)*260*260*4 <= 12*10^6 implies TS <= 9.0947

Comparing with the naive code

Time-skewing: 2 loads + 2 stores produces TS iterations
Naive: 2*TS loads + 2*TS stores produces TS iterations

Results

Run on DAVinCI(exclusive node)
Grid size: 260*260*260
Total time steps: 27721
Data Type: float
Computing Difference:
TS~2.6e-8, SIMD~5.19e-7
(naive: 0.0013~ -0.0015)



Future Research:

1. Perform experiments with stencil codes of higher order (Finite difference scheme with higher spatial error order)

2. Work out a method to deduce the optimal tile size and time step without the need to do the experiments

3. Integrate onto IWAVE package to enhance its computation capability of the stencil kernel.

4. Modifications for the codes if the coefficients are not constant or changing between two time steps.

5. ...

Thanks! Suggestions?