BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

# THE PEBBLE GAME

## (as a Computational Model)

Author: **Ing. Jakub Budiský**, xbudis02@stud.fit.vutbr.cz

Dec 7, 2016

# Games in Theoretical Computer Science

- Good as models
  - Simplified representation of a real system
  - Usually provide a “good level of abstraction”
- Simple rules allow for better understanding
  - Easy to work with

# The (Black) Pebble Game

- Played on a directed acyclic graph (DAG)
- Playing
  - Placing pebbles on vertices of a graph
  - Starting with no pebbles placed
  - Complying with a given set of rules

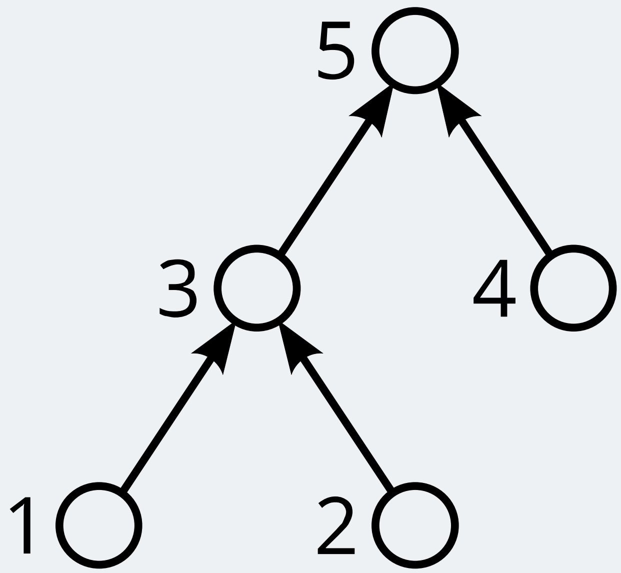
# The (Black) Pebble Game

- Rules
  - A pebble can be placed on an input vertex at any time
  - A pebble can be placed on (moved to) any non-input vertex only if all its immediate predecessors carry pebbles
  - A pebble can be removed at any time
  - Each output vertex must be pebbled at least once

# What are we modeling

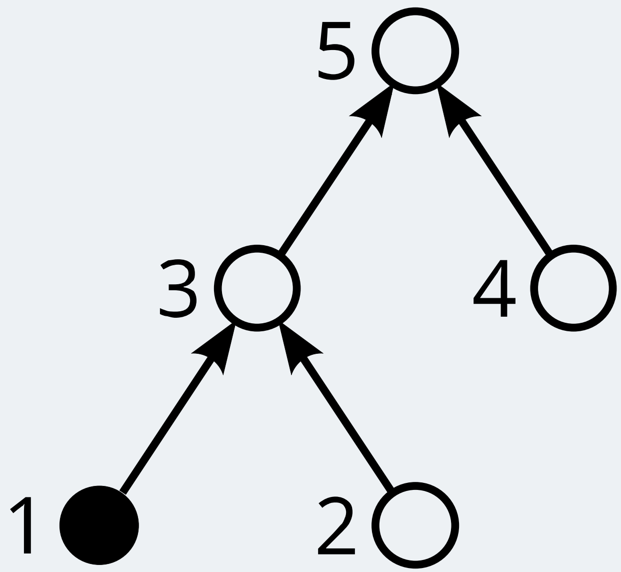
- DAG = data dependencies
- Vertices = computed data
  - Input vertices are inputs, output vertices results
- Pebbles = registers
- Pebbling = computational step
  - Removing pebbles is usually neglected

# Example



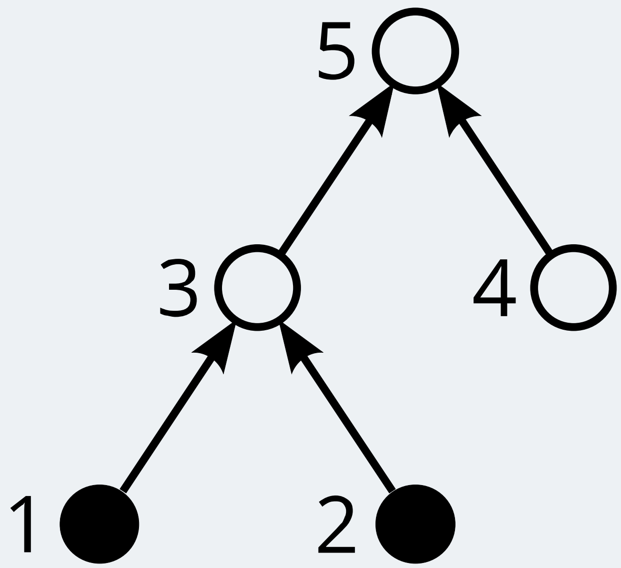

$\{\}$

# Example



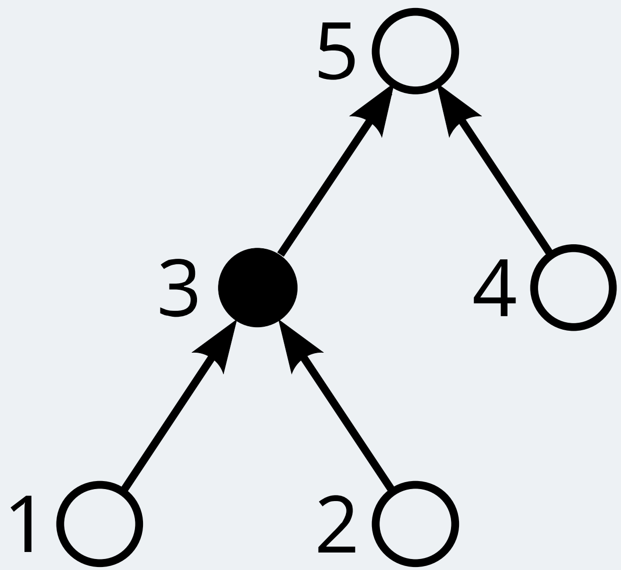

$$\{\} \underset{1}{\Rightarrow} \{1\}$$

# Example




$$\{\} \underset{1}{\Rightarrow} \{1\} \underset{2}{\Rightarrow} \{1, 2\}$$

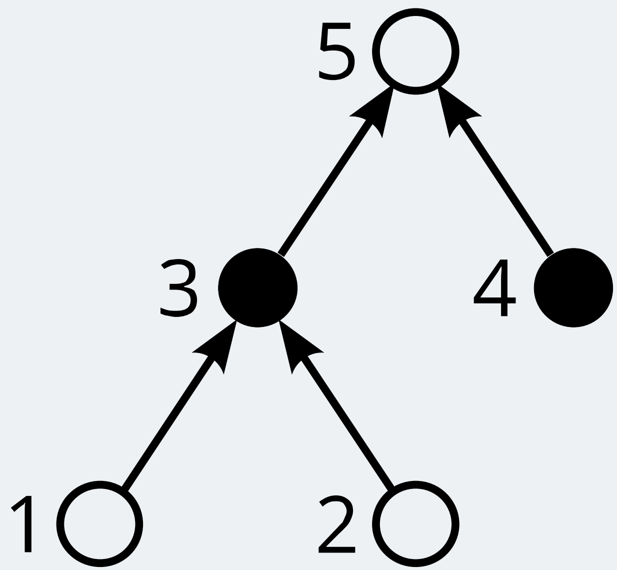
# Example




$$\{\} \underset{1}{\Rightarrow} \{1\} \underset{2}{\Rightarrow} \{1,2\} \underset{2}{\Rightarrow}$$

$$\underset{2}{\Rightarrow} \{3\}$$

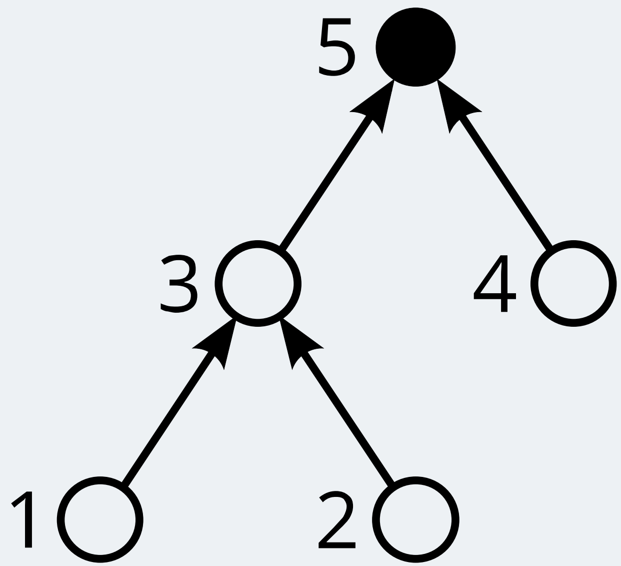
# Example




$$\{\} \underset{1}{\Rightarrow} \{1\} \underset{2}{\Rightarrow} \{1, 2\} \underset{2}{\Rightarrow}$$

$$\underset{2}{\Rightarrow} \{3\} \underset{2}{\Rightarrow} \{3, 4\}$$

# Example




$$\{\} \underset{1}{\Rightarrow} \{1\} \underset{2}{\Rightarrow} \{1, 2\} \underset{2}{\Rightarrow}$$

$$\underset{2}{\Rightarrow} \{3\} \underset{2}{\Rightarrow} \{3, 4\} \underset{2}{\Rightarrow} \{5\}$$

# Limitations of The (Black) Pebble Game

- Limited to straight-line programs
  - FFT, matrix multiplication, convolution,...
  - If the branching is bounded, we can use a multi-graph
    - Otherwise you need a better model
    - Sorting, merging, compression,...
- Less powerful than other models
  - Turing machines, RAMs,...

# Why we still love it

- Problems that can be interpreted using the (black) pebble game are important
- Simplicity
- Close to real computers
- Extensible

# What is it good for

- Analyzing complexity classes
- Investigating space-time trade-offs
  - DAG transformations are not performed
- Optimization of database queries
- Proofs that algorithms are optimal
  - In terms of a “pebbling strategy”

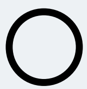
# Investigating space-time trade-offs

- Number of computational steps (pebbling) required = time complexity
  - Optimizing this parameter means optimizing data reuse (“single-threaded” programs)
- Number of pebbles required = space complexity

# Typical (sub)graphs

- Complete binary tree
- Pyramid
- Butterfly

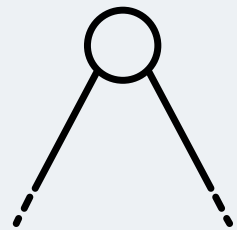
# Complete binary tree



$T(0)$

$S_{min}(T(0)) = 1$

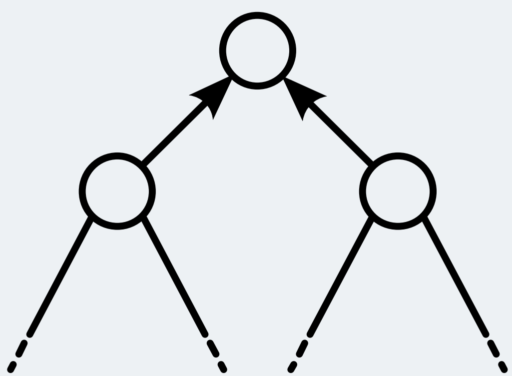
1 step



$T(k-1)$

$S_{min}(T(k-1)) = k$

$2^k - 1$ steps

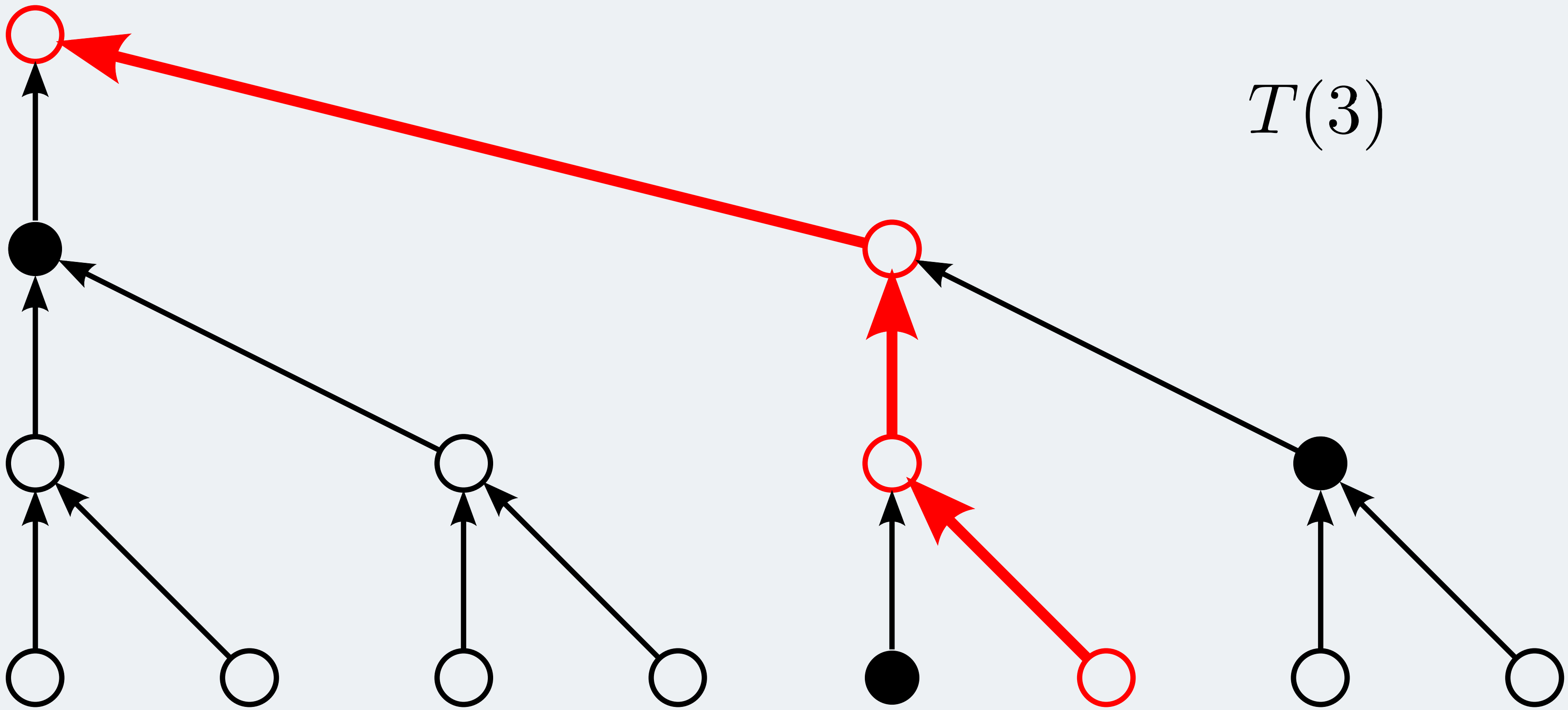


$T(k)$

$S_{min}(T(k)) = k + 1$

$2(2^k - 1) + 1 = 2^{k+1} - 1$ steps

# Complete binary tree

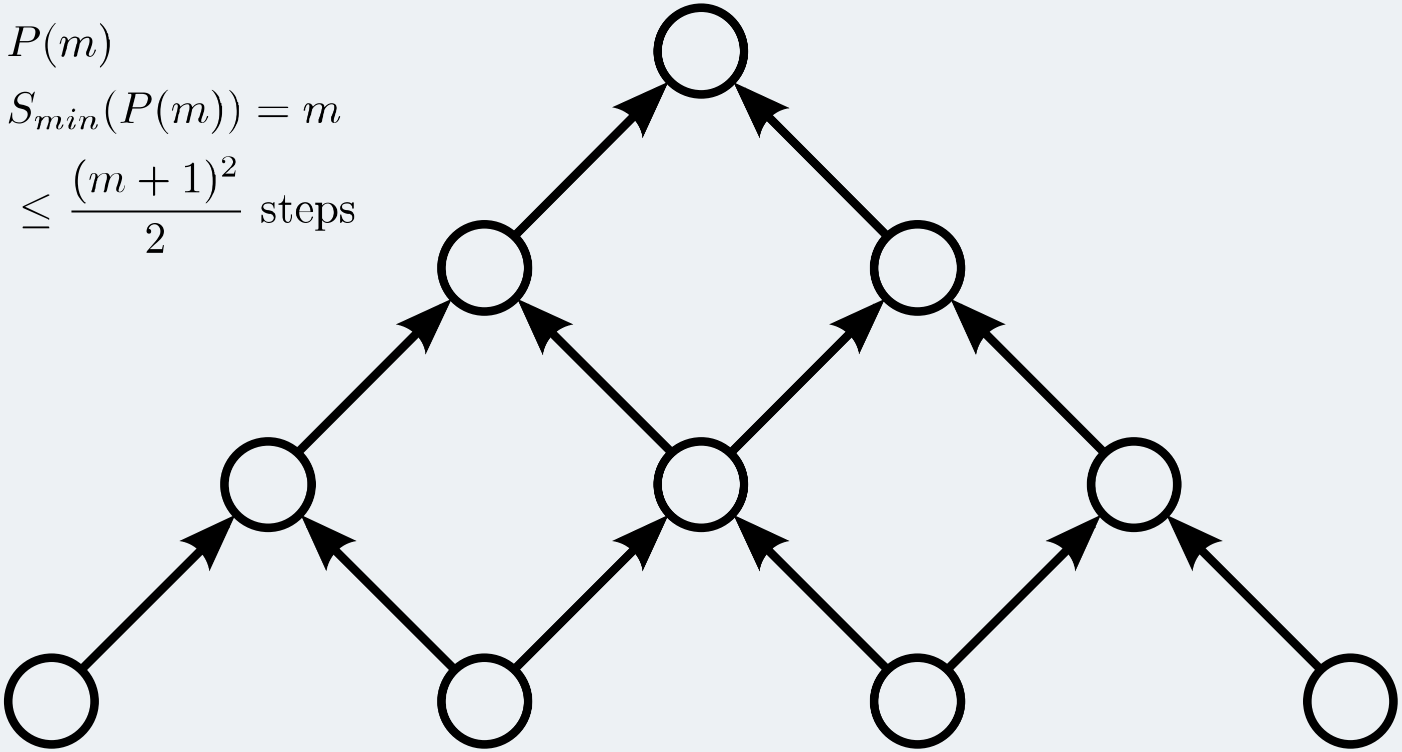


# Pyramid

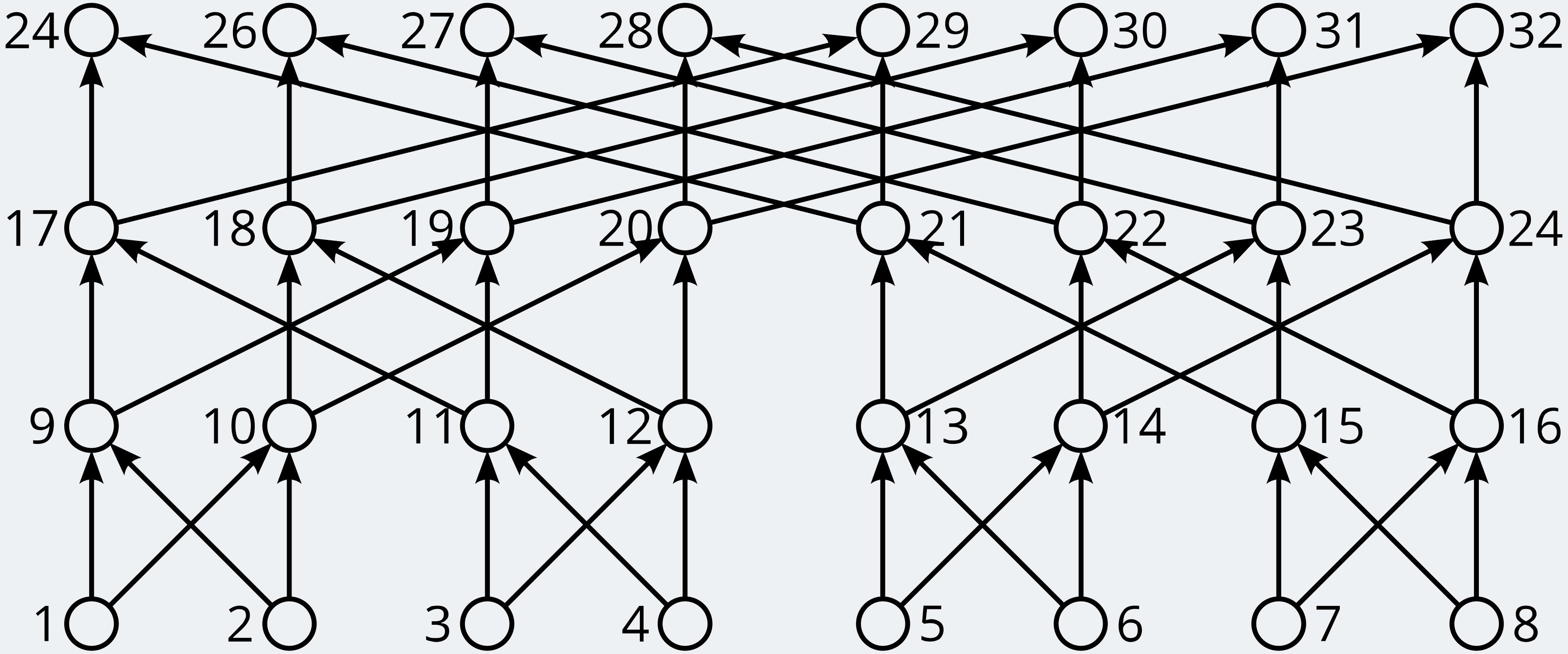
$P(m)$

$S_{min}(P(m)) = m$

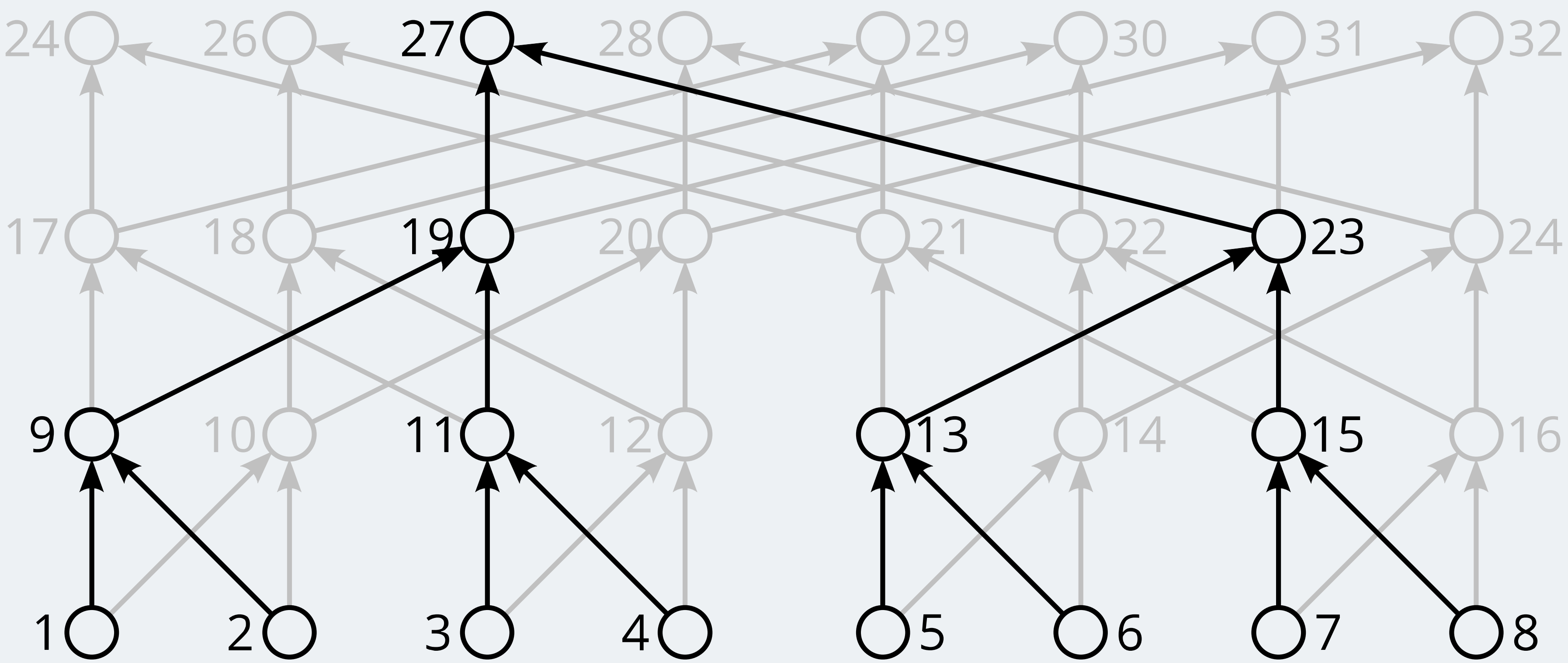
$\leq \frac{(m+1)^2}{2}$ steps



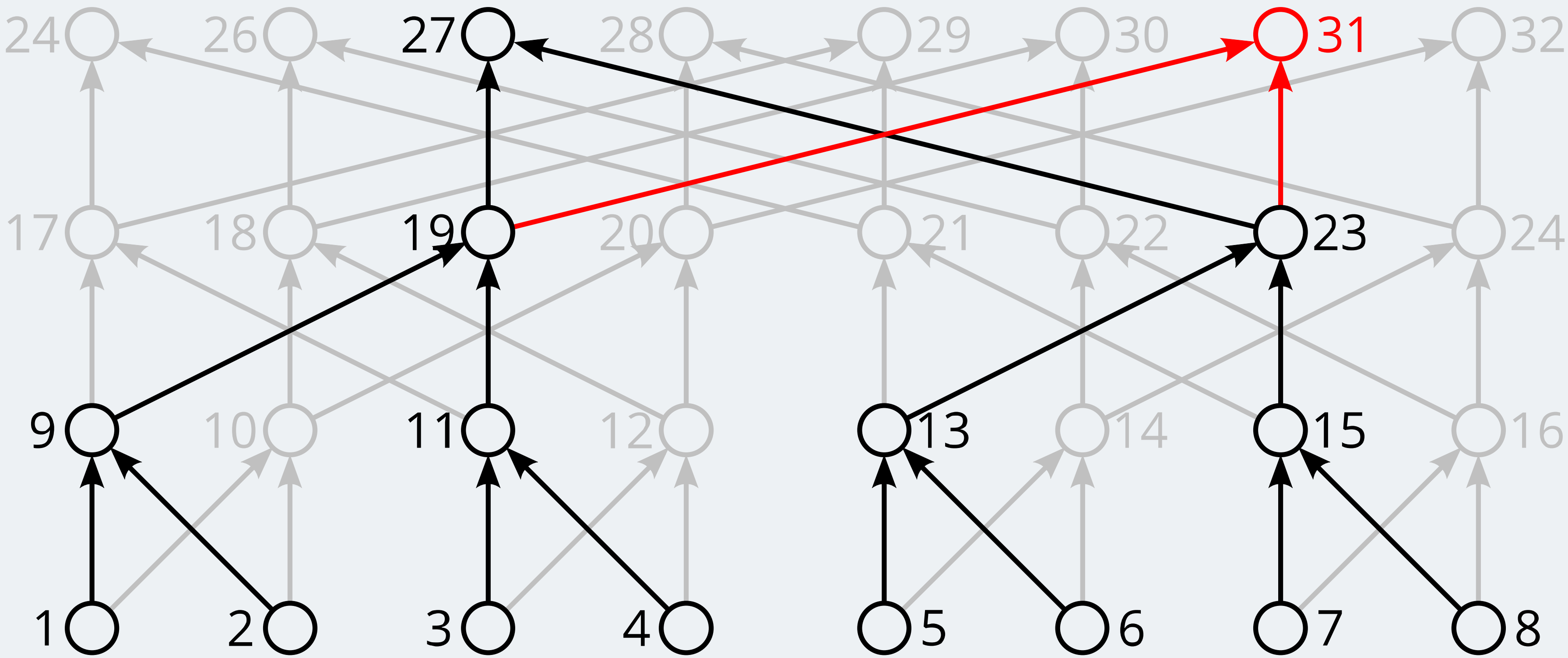
# FFT butterfly



# FFT butterfly



# FFT butterfly



# Further notes

- Families of graphs with extreme trade-offs exist
  - One more pebble can reduce the time complexity from exponential to polynomial
- Proving optimal solutions for more complex functions
  - Grigoriev's lower bound method
    - Classifying functions based on their "flow property"
    - (S+1)T complexity

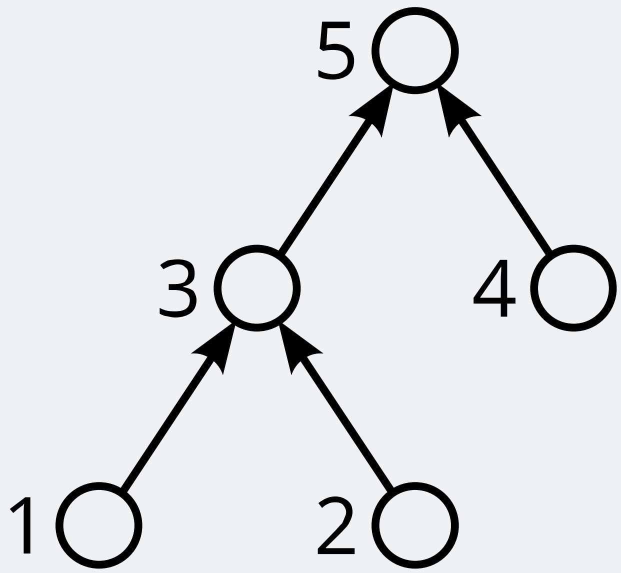
# Possible extensions

- Non-determinism
- Adding an interpretation to the DAG
- Introducing another player
- Better approximation of real-life hardware
- …

# Black-White Pebble Game

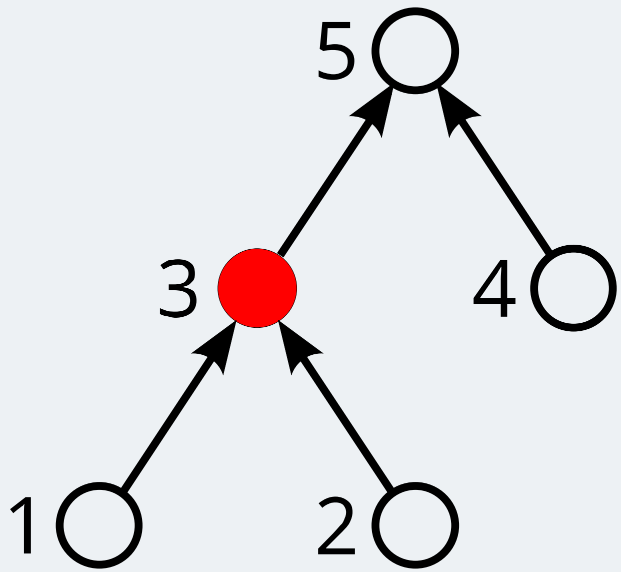
- White pebbles are used to express a “guess” that is justified later
- Models a non-deterministic computation
- Non-determinism can save space on some graphs

# Example



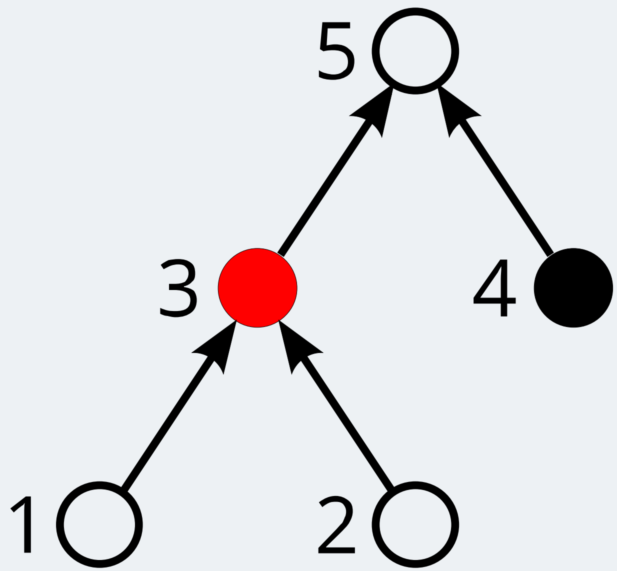

$$(\{\}, \{\})$$

# Example



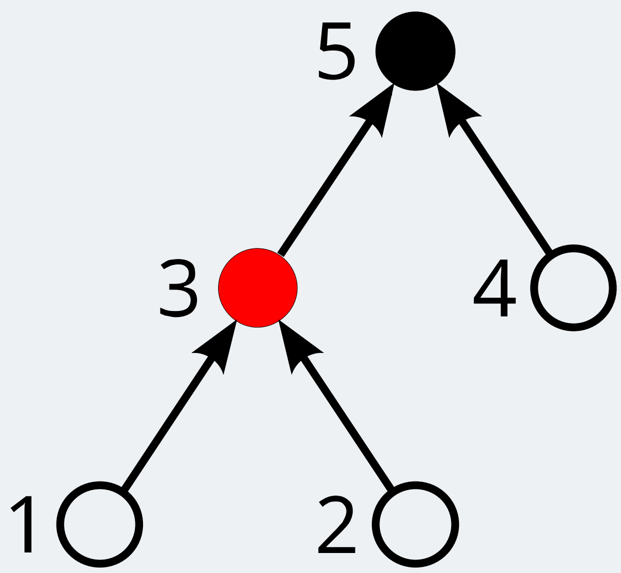

$$(\{\}, \{\}) \underset{1}{\Rightarrow} (\{\}, \{3\})$$

# Example



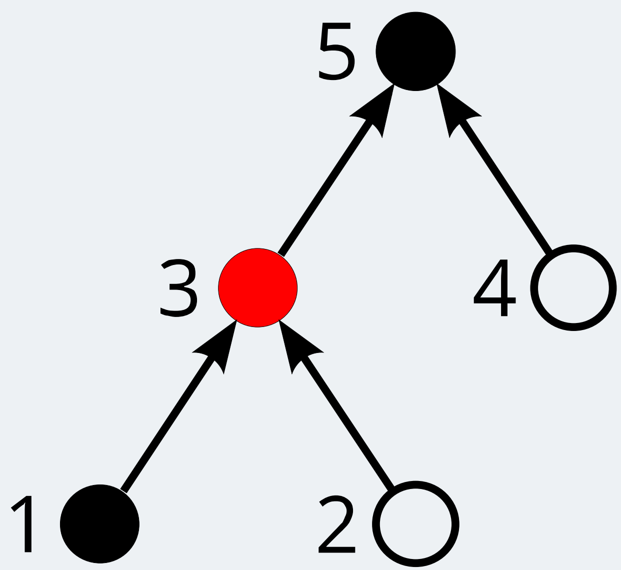

$$(\{\}, \{\}) \underset{1}{\Rightarrow} (\{\}, \{3\}) \underset{2}{\Rightarrow}$$
$$\underset{2}{\Rightarrow} (\{4\}, \{3\})$$

# Example



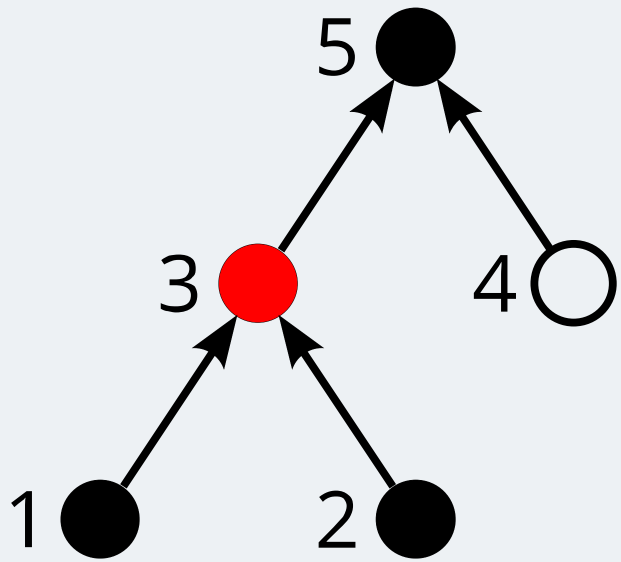

$$(\{\},\{\}) \underset{1}{\Rightarrow} (\{\},\{3\}) \underset{2}{\Rightarrow}$$
$$\underset{2}{\Rightarrow} (\{4\},\{3\}) \underset{2}{\Rightarrow} (\{5\},\{3\})$$

# Example



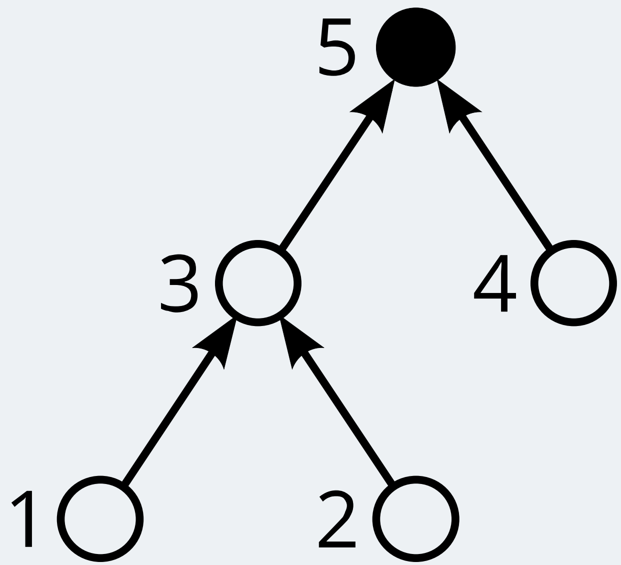

$$(\{\}, \{\}) \underset{1}{\Rightarrow} (\{\}, \{3\}) \underset{2}{\Rightarrow}$$
$$\underset{2}{\Rightarrow} (\{4\}, \{3\}) \underset{2}{\Rightarrow} (\{5\}, \{3\}) \underset{3}{\Rightarrow}$$
$$\underset{3}{\Rightarrow} (\{5, 1\}, \{3\})$$

# Example




$$(\{\}, \{\}) \underset{1}{\Rightarrow} (\{\}, \{3\}) \underset{2}{\Rightarrow}$$
$$\underset{2}{\Rightarrow} (\{4\}, \{3\}) \underset{2}{\Rightarrow} (\{5\}, \{3\}) \underset{3}{\Rightarrow}$$
$$\underset{3}{\Rightarrow} (\{5, 1\}, \{3\}) \underset{4}{\Rightarrow}$$
$$\underset{4}{\Rightarrow} (\{5, 1, 2\}, \{3\})$$

# Example




$$(\{\}, \{\}) \underset{1}{\Rightarrow} (\{\}, \{3\}) \underset{2}{\Rightarrow}$$
$$\underset{2}{\Rightarrow} (\{4\}, \{3\}) \underset{2}{\Rightarrow} (\{5\}, \{3\}) \underset{3}{\Rightarrow}$$
$$\underset{3}{\Rightarrow} (\{5, 1\}, \{3\}) \underset{4}{\Rightarrow}$$
$$\underset{4}{\Rightarrow} (\{5, 1, 2\}, \{3\}) \underset{4}{\Rightarrow} (\{5\}, \{\})$$

# Red-Blue Pebble Game

- Modeling hierarchical memory (I/O)
- Hot (red) pebbles represent main memory
- Blue pebbles represent a secondary storage
- Computation is done with red pebbles

# Two-person Pebble Game

- Several variants
  - Some of them proven equivalent later
  - A lot of variety
- Usually interpreted graphs and defined relations to other models
  - Added capabilities and complexity
- Roles may and may not be symmetric

# Two-person Pebble Game

- Example: Venkateswaran and Tompa
  - Symmetric player roles
  - Boolean circuits (affects rules)
  - Characterizes LOGCFL and AC
  - It is possible to prove Savitch's theorem
  - …

# Conclusions

- What is the Black Pebble Game ("basic one")
- What are the advantages of the model
- What are some of the limitations
- Basic examples
- Where you can get with it if you want(ed) to

# References


- Space-Time Tradeoffs. 2008. SAVAGE, J. E. *Models of Computation: Exploring the Power of Computing* [online]. p. 462-488. Available at: http://cs.brown.edu/~jes/book/
- VENKATESWARAN, H. and M. TOMPA. 1989. A New Pebble Game that Characterizes Parallel Complexity Classes. *SIAM Journal on Computing.* 18(3), 533-549.
- MEYER AUF DER HEIDE, F. 1981. A comparison of two variations of a pebble game on graphs. *Theoretical Computer Science.* 13(3), 315-322.