Representing integer multiplication using BDDs

Håvard Raddum and Srimathi Varadharajan
Simula@UiB

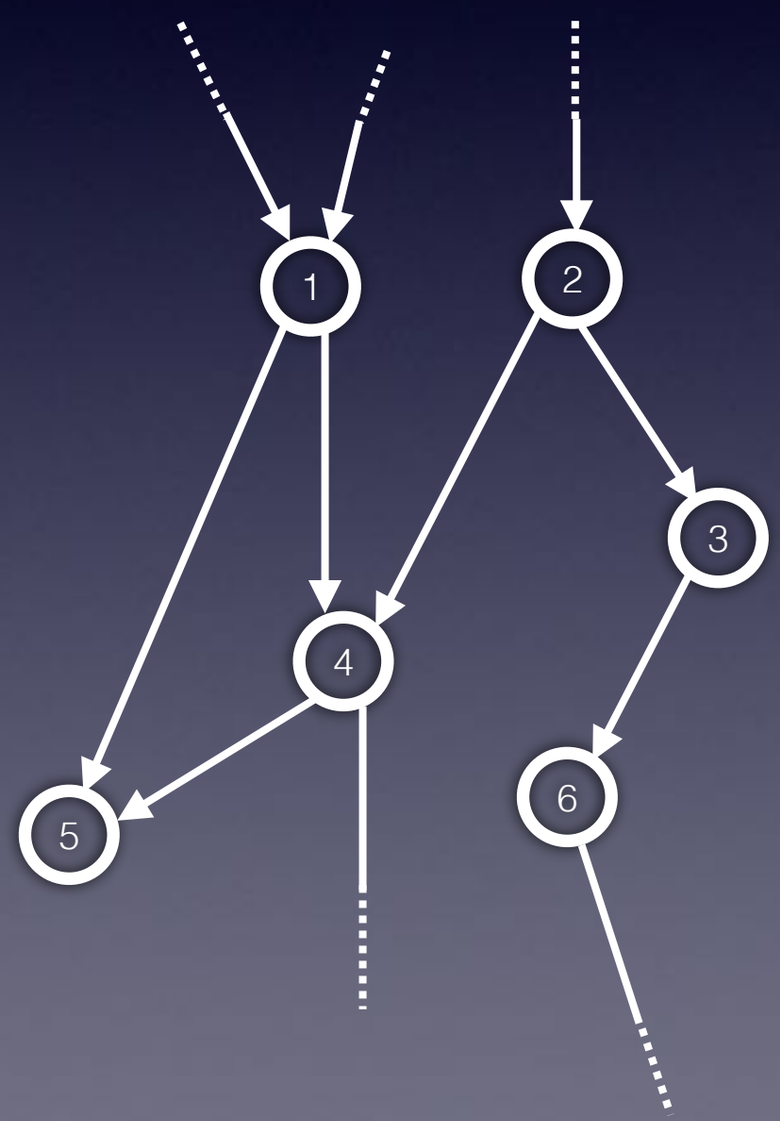
What is a BDD?

Can be defined and understood in different ways

Only one description given here

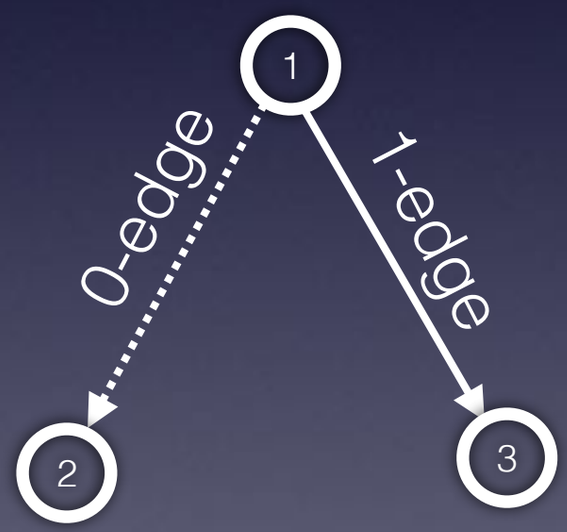
BDD explained

A BDD is a directed acyclic graph, drawn from top to bottom



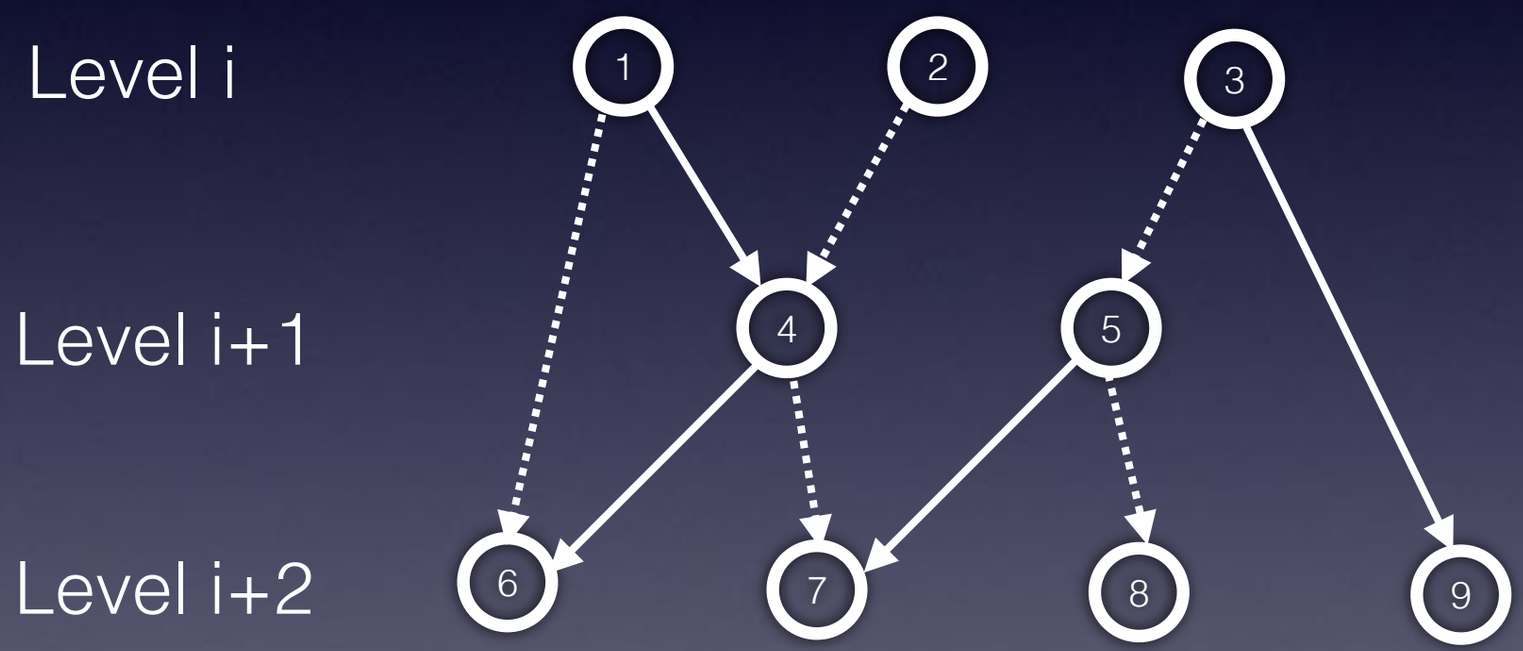
BDD explained

Every node can have two outgoing edges,
the 0-edge and the 1-edge



BDD explained

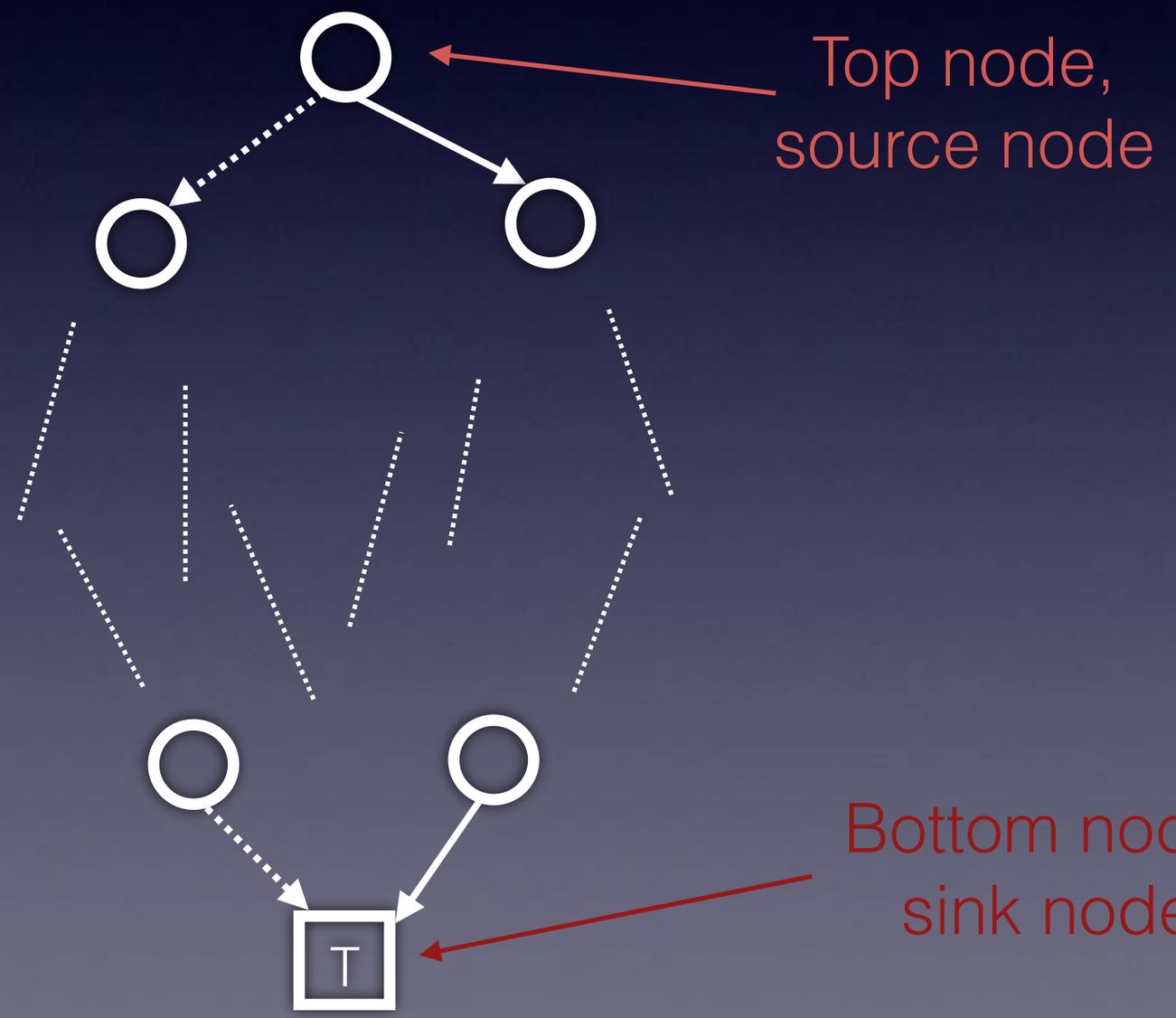
Nodes are arranged in horizontal levels



No edges between nodes on same level

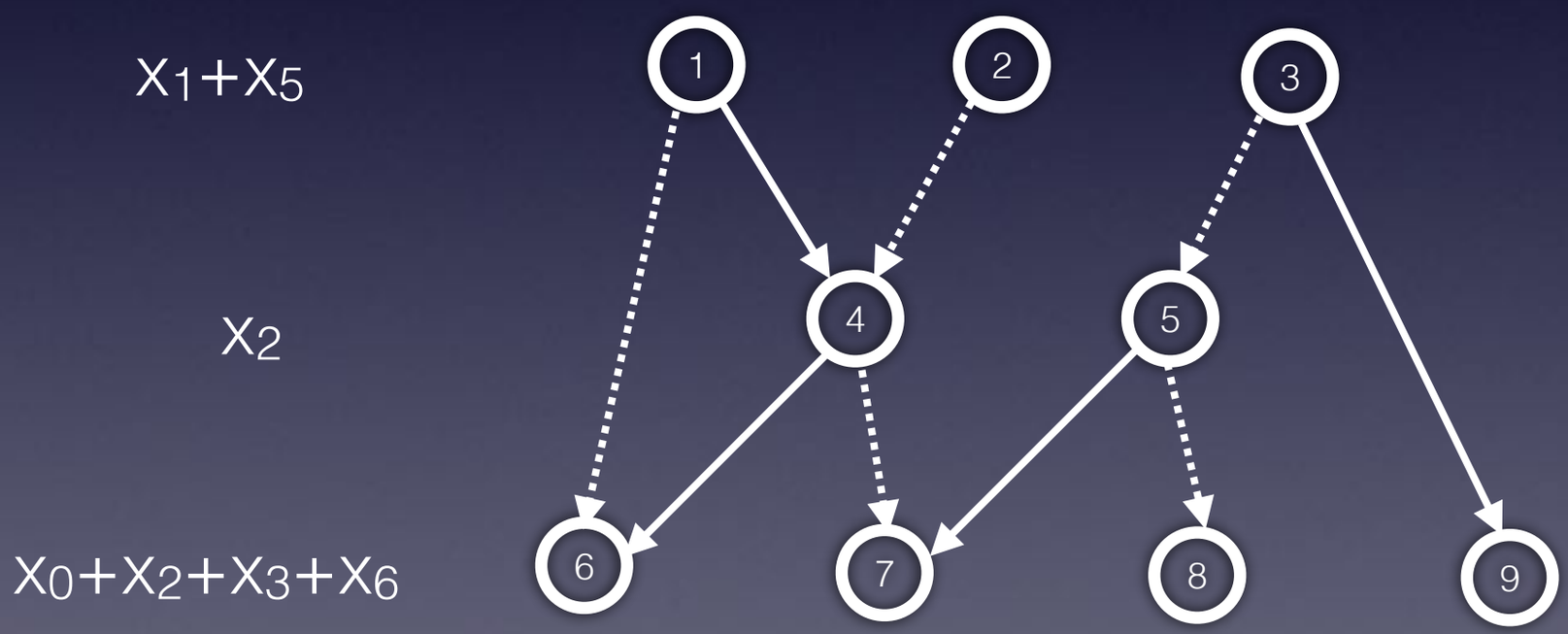
BDD explained

Only one node on highest and lowest levels



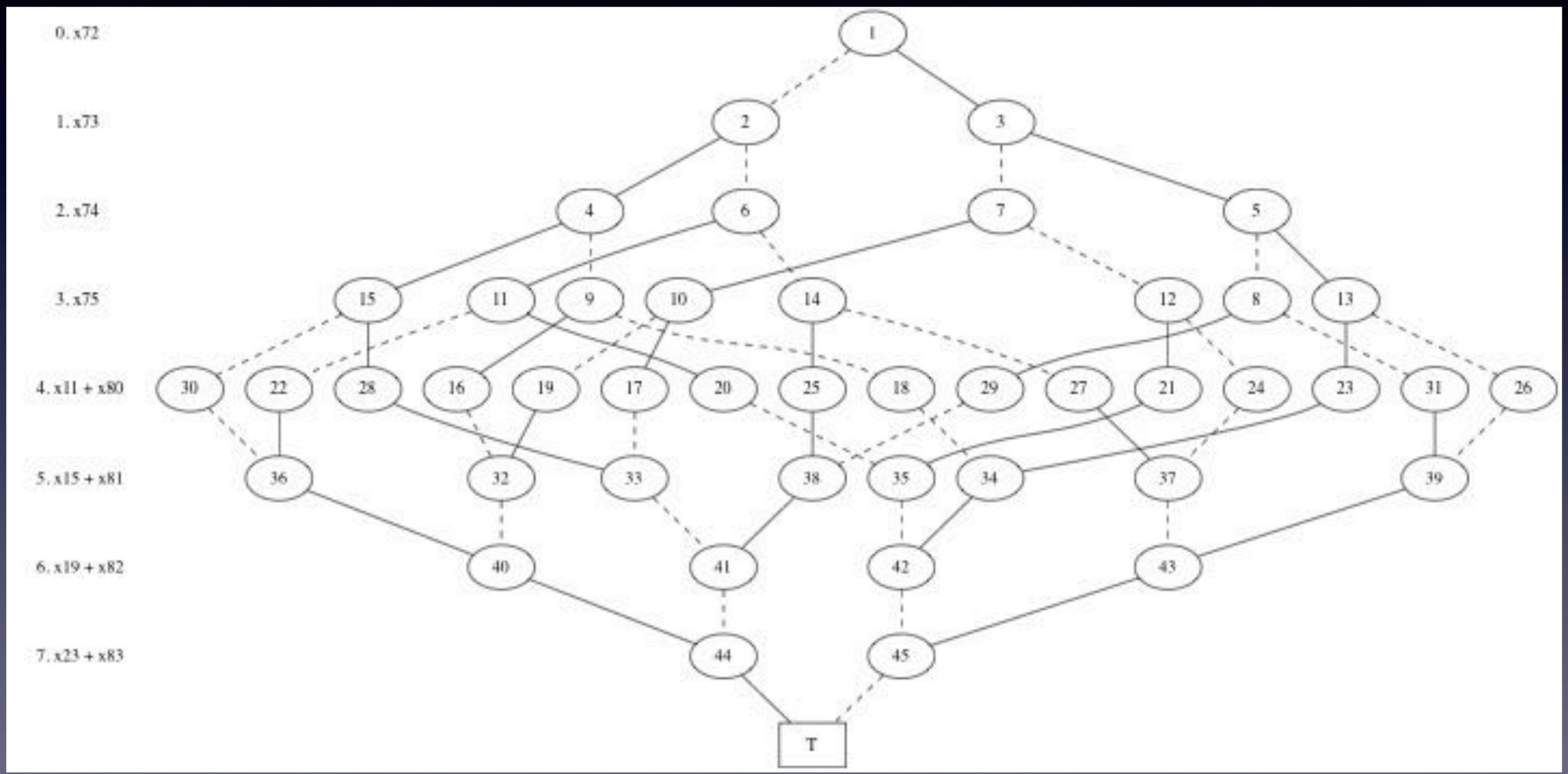
BDD explained

Linear combination of variables associated with each level, except bottom level

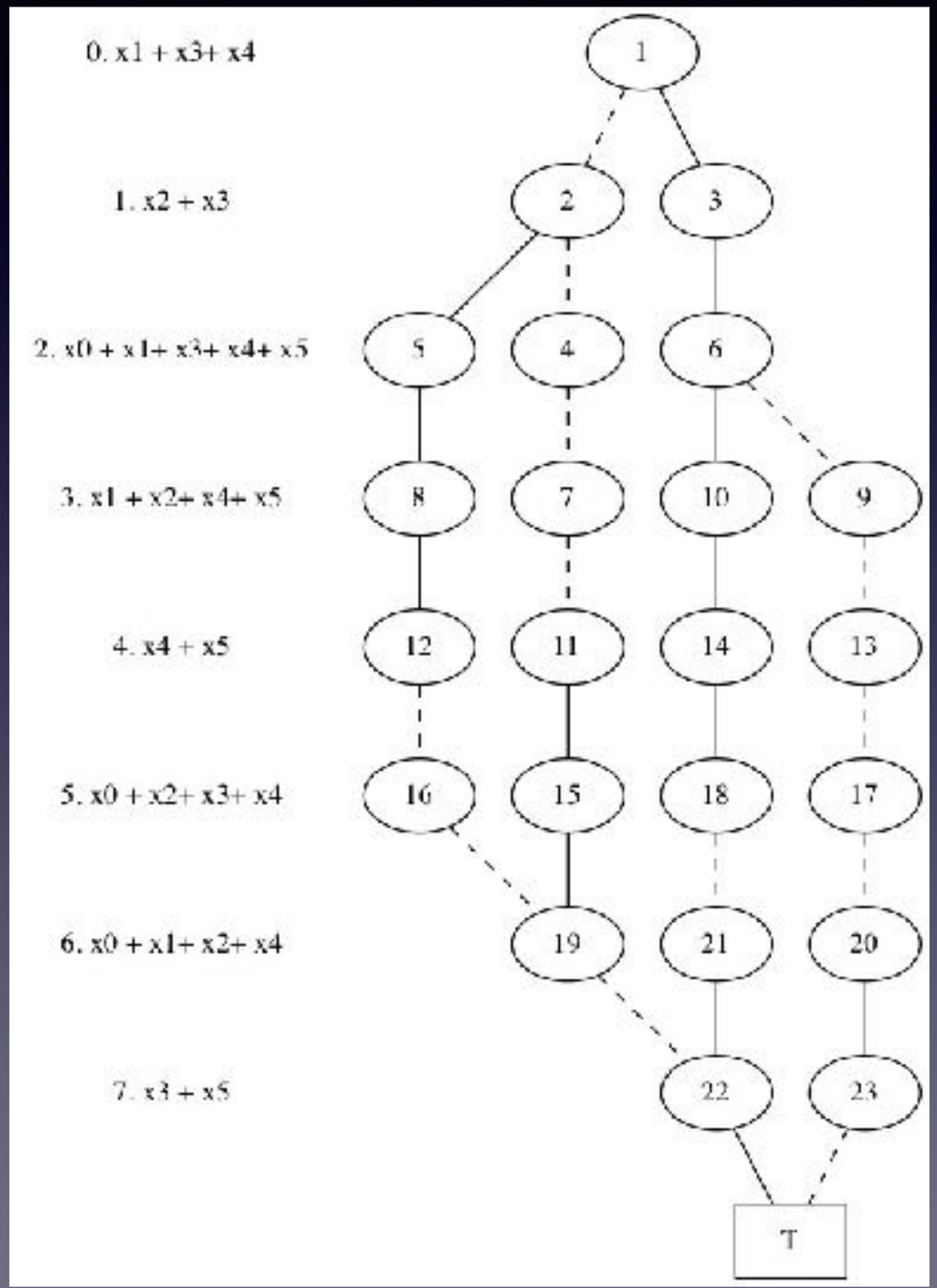


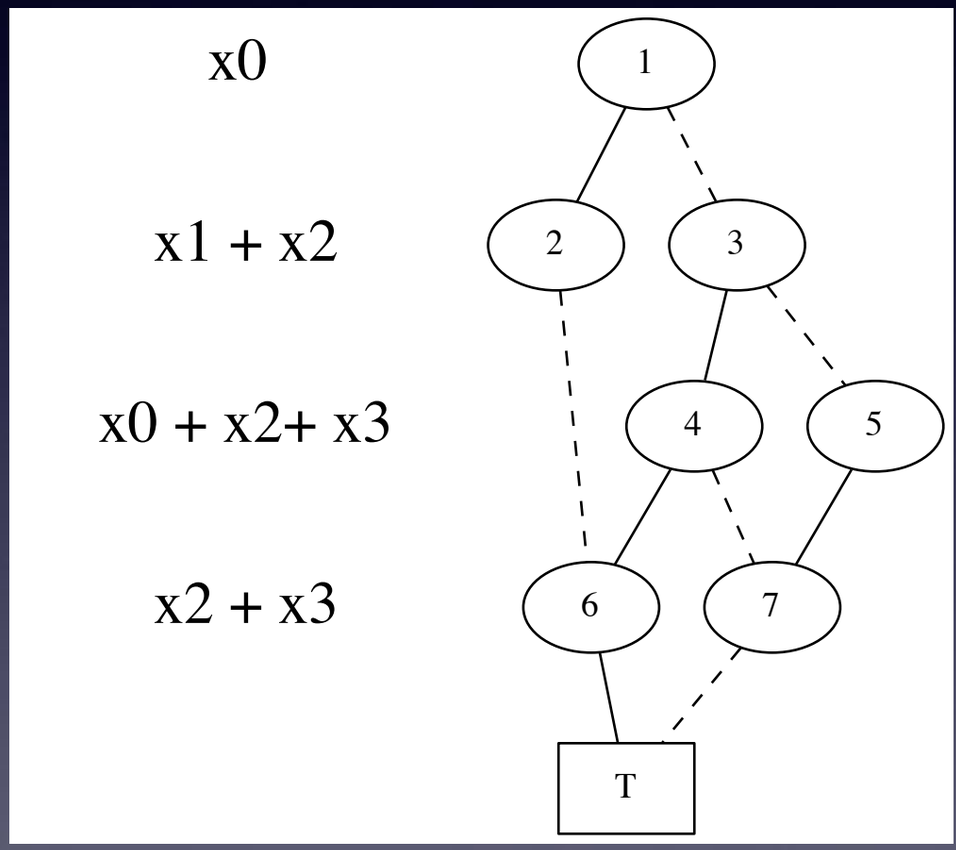
Other descriptions: only single variables associated with levels

BDD Examples

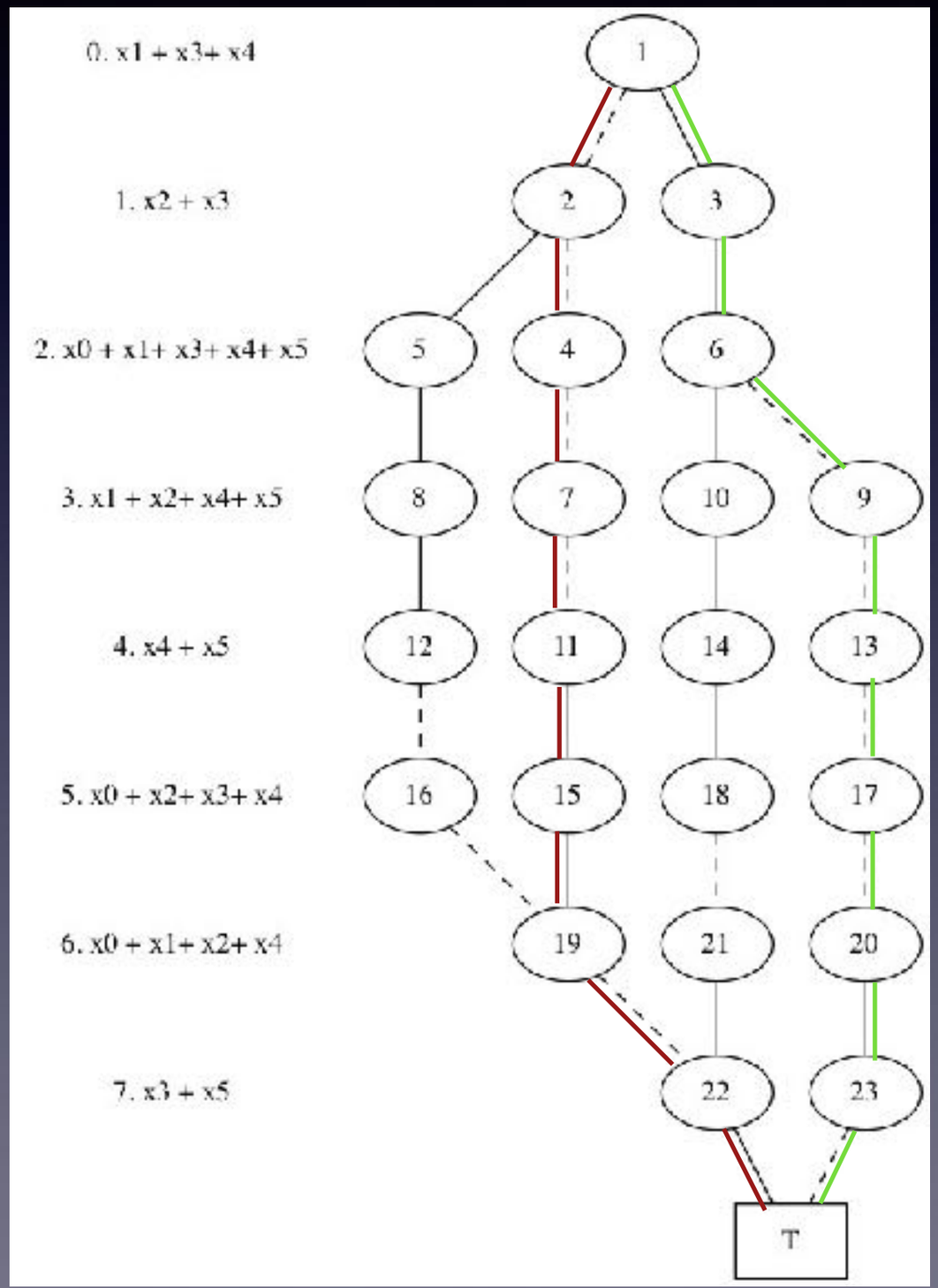


BDD examples





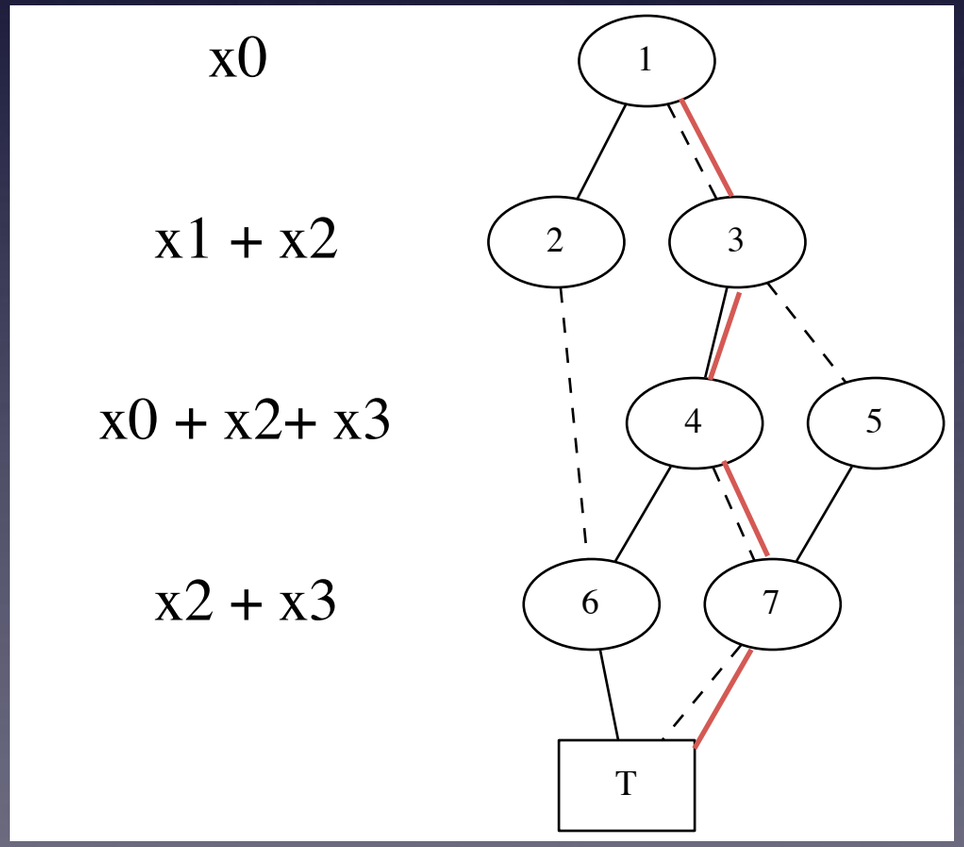
Paths



0	1
0	1
0	0
0	0
1	0
1	0
0	1
1	0

Assigning values

Values in path assigned to linear combinations associated with levels



$$x_0 = 0$$

$$x_1 + x_2 = 1$$

$$x_0 + x_2 + x_3 = 0$$

$$x_2 + x_3 = 0$$

Integer multiplication as BDD

How to construct multiplication BDD?

N, p and q in binary

- RSA modulus $N=pq$
- $p=p_{n-1}\, p_{n-2} \ldots\ p_2\, p_1\, p_0$
- $q=q_{n-1}\, q_{n-2} \ldots\ q_2\, q_1\, q_0$
- $N=N_{2n-1}\, N_{2n-2} \ldots\ N_2\, N_1\, N_0$
- $N_i, p_i, q_i \in \{0,1\}$

Multiplication

	$(q_{n-1}$	$\cdots$	q_2	q_1	$q_0)$	$(p_{n-1}$	$\cdots$	p_2	p_1	$p_0)$
						p_0q_{n-1}	$\cdots$	p_0q_2	p_0q_1	p_0q_0
$+$					p_1q_{n-1}	p_1q_{n-2}	$\cdots$	p_1q_1	p_1q_0	
$+$				p_2q_{n-1}	p_2q_{n-2}	p_2q_{n-3}	$\cdots$	p_2q_0		
$\vdots$			$\iddots$	$\vdots$	$\vdots$	$\vdots$	$\iddots$			
$+$		$p_{n-1}q_{n-1}$	$\cdots$	$p_{n-1}q_2$	$p_{n-1}q_1$	$p_{n-1}q_0$				
$=$	N_{2n-1}	N_{2n-2}	$\cdots$	N_{n+1}	N_n	N_{n-1}	$\cdots$	N_2	N_1	N_0
		2n-2		n+1	n	n-1		2	1	0

columns

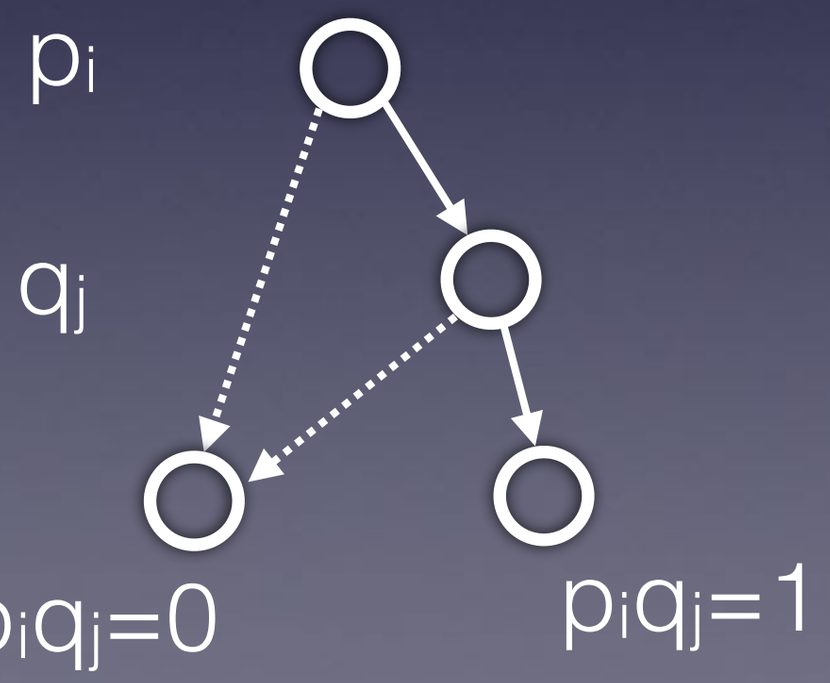
Column k consists of all terms p_iq_j where $i+j=k$

Column k contributes $2^k\sum p_iq_j$ to the value of N

Building multiplication BDD

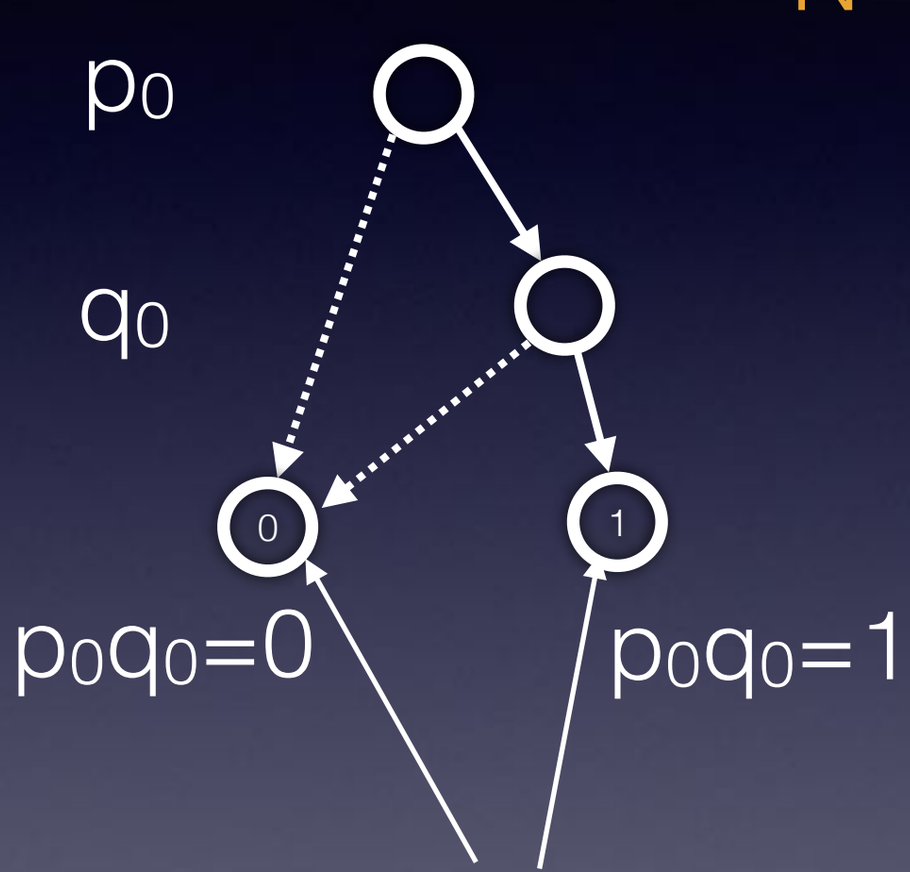
- BDD built column by column, starting with column 0
- One level of nodes constructed from each variable p_i, q_j appearing in column k

Basic building block:

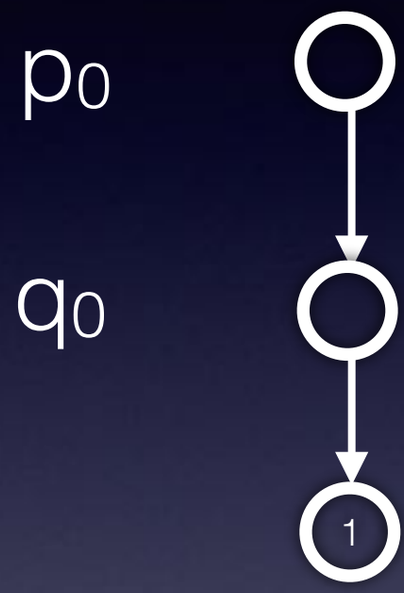


Column 0

N=…011



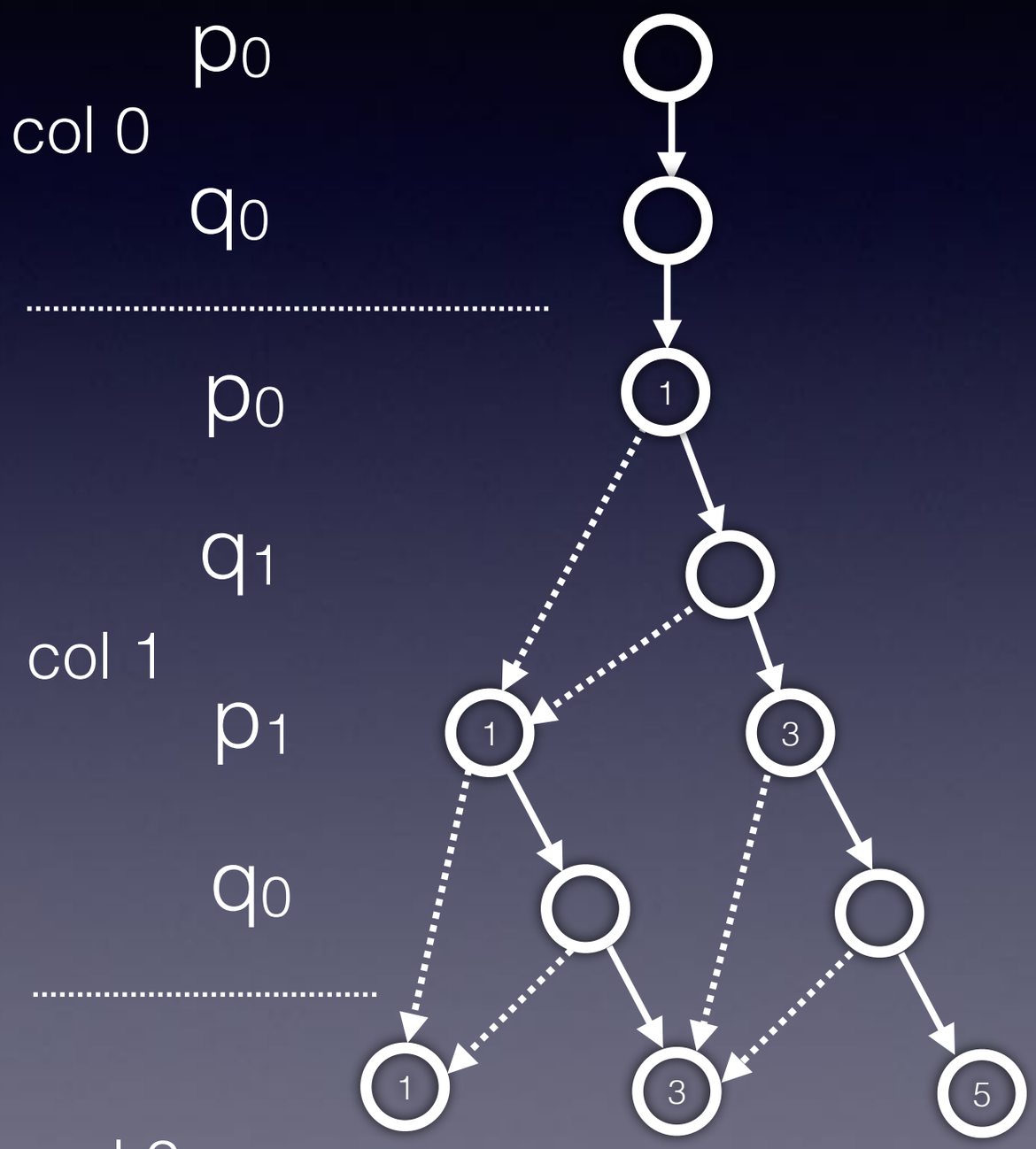
keep track of value of N computed so far

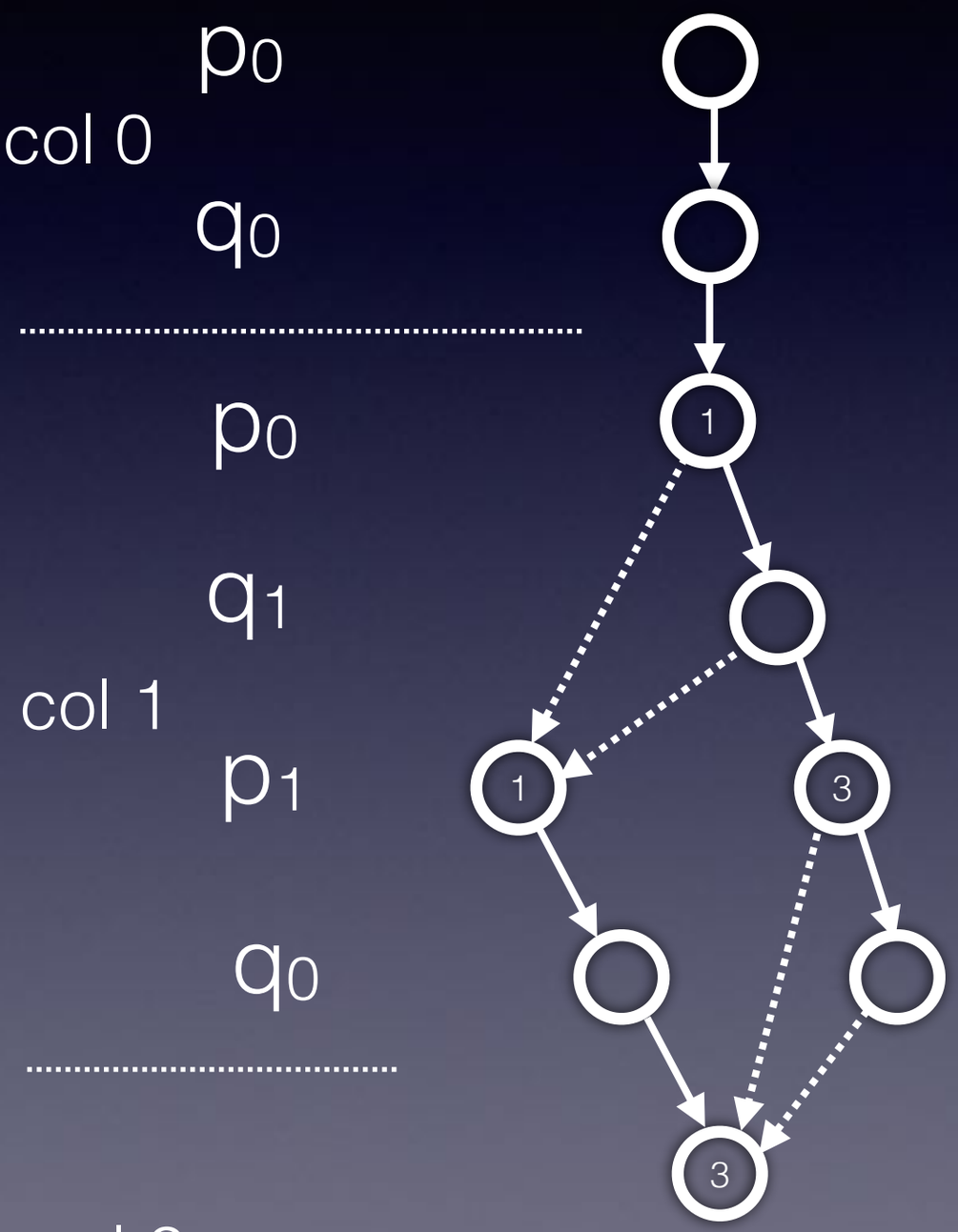


When construction of column k is complete, delete nodes with values inconsistent with N_k

Column 1

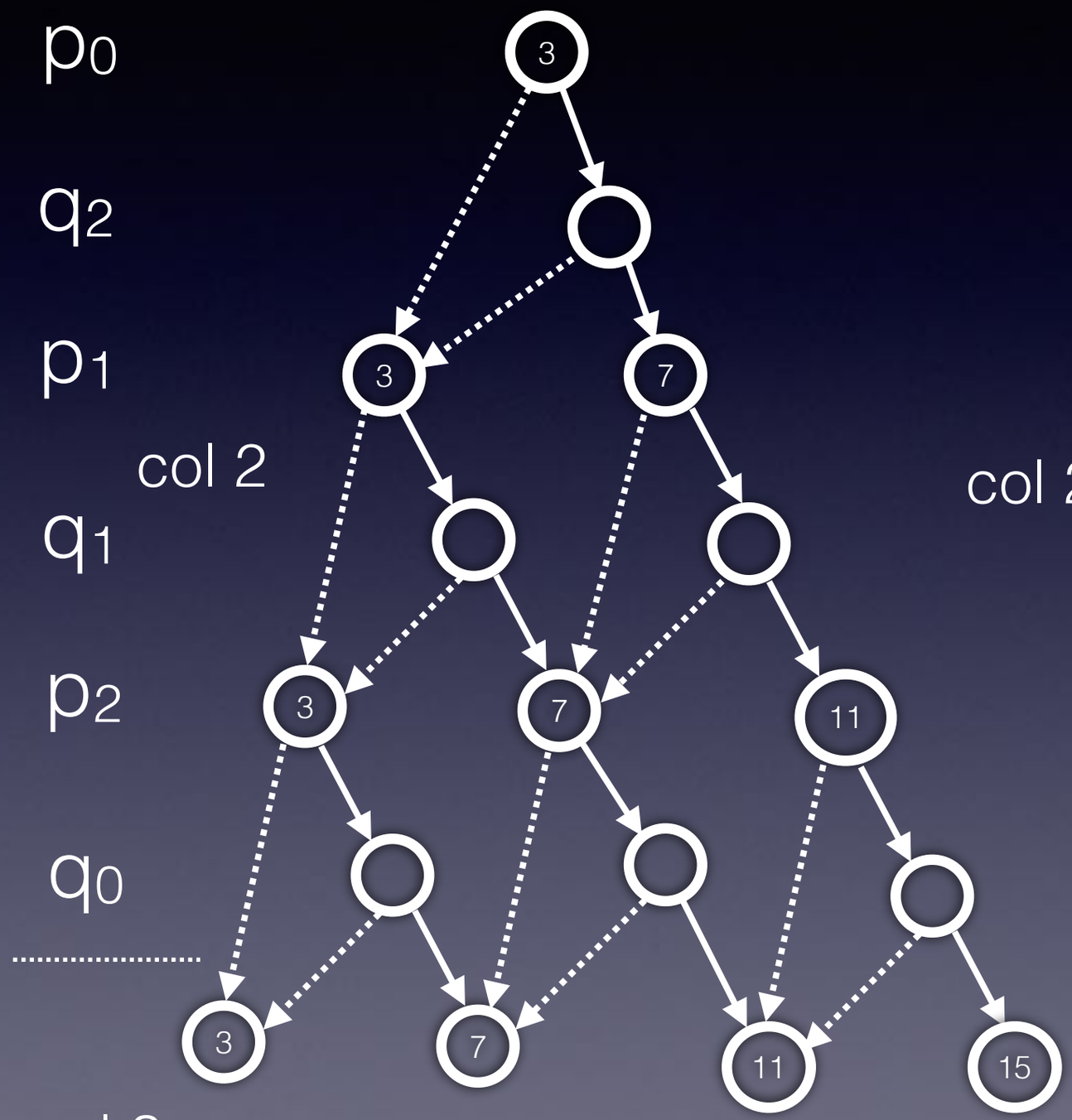
N=…011

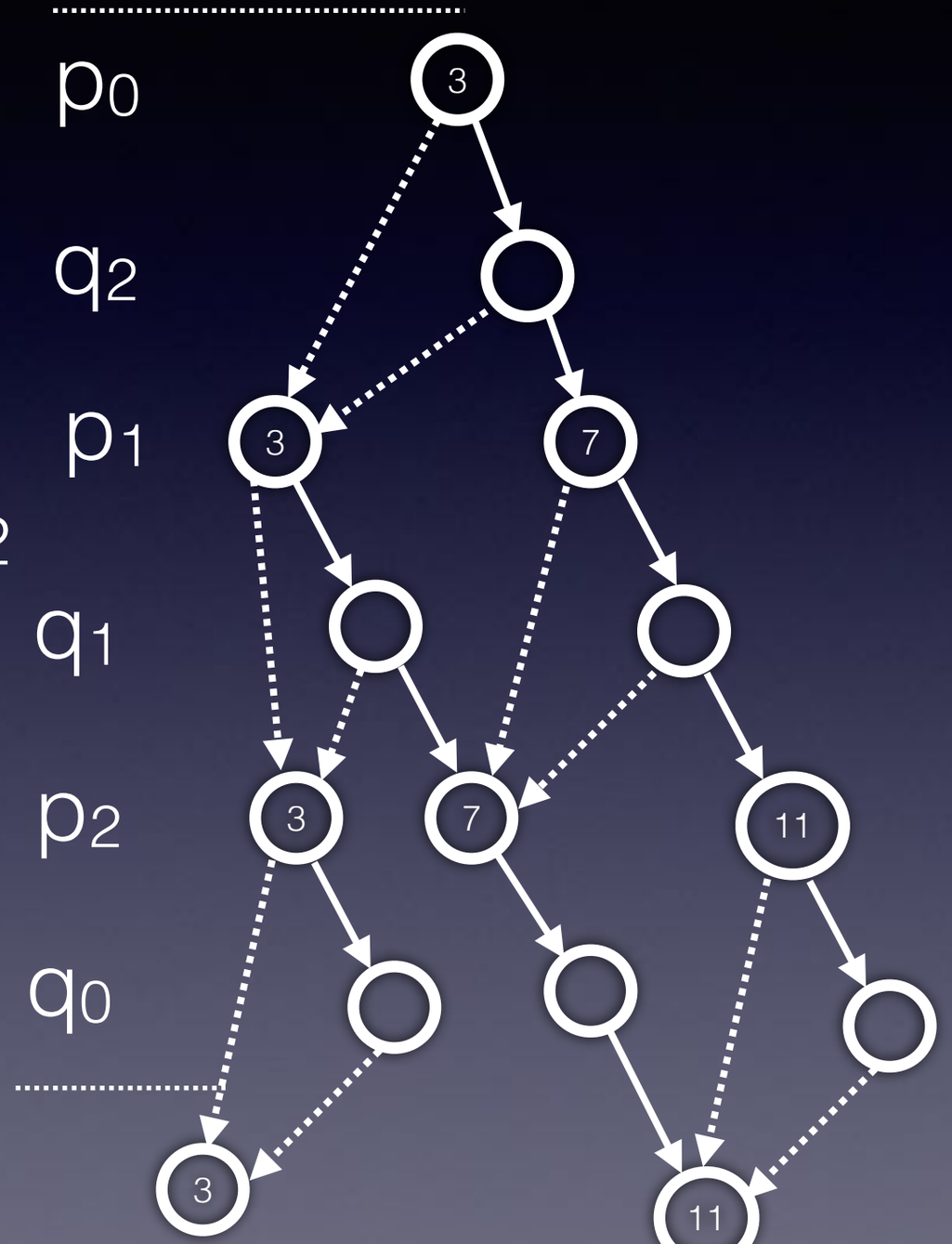




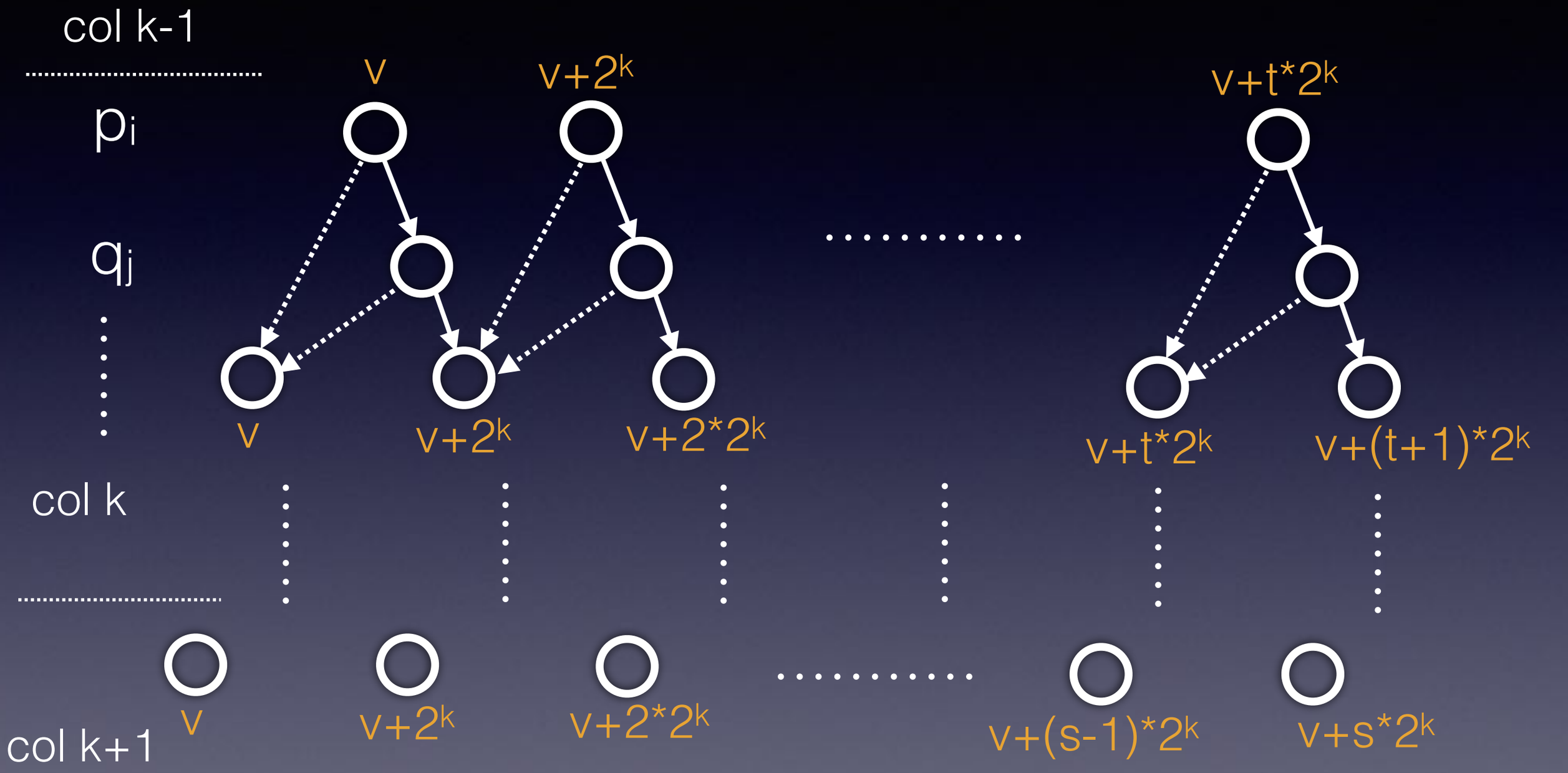
Column 2

N=...011

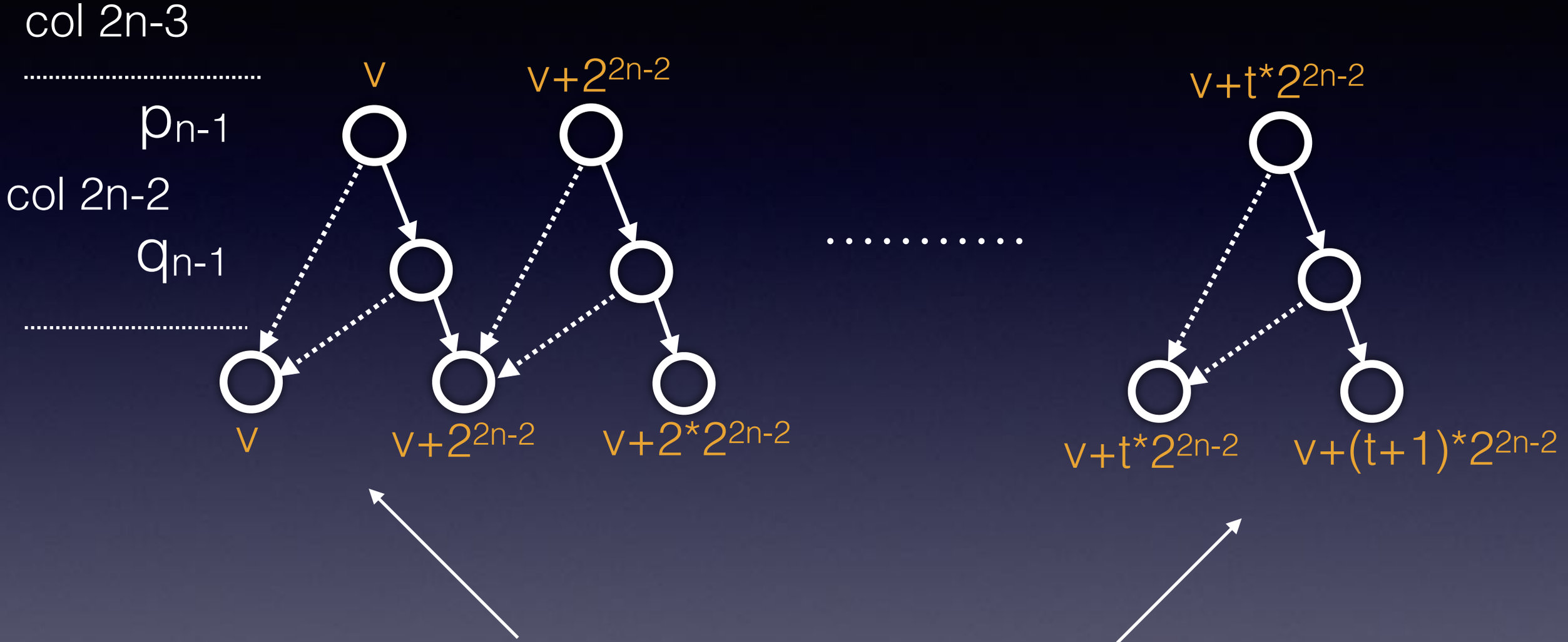




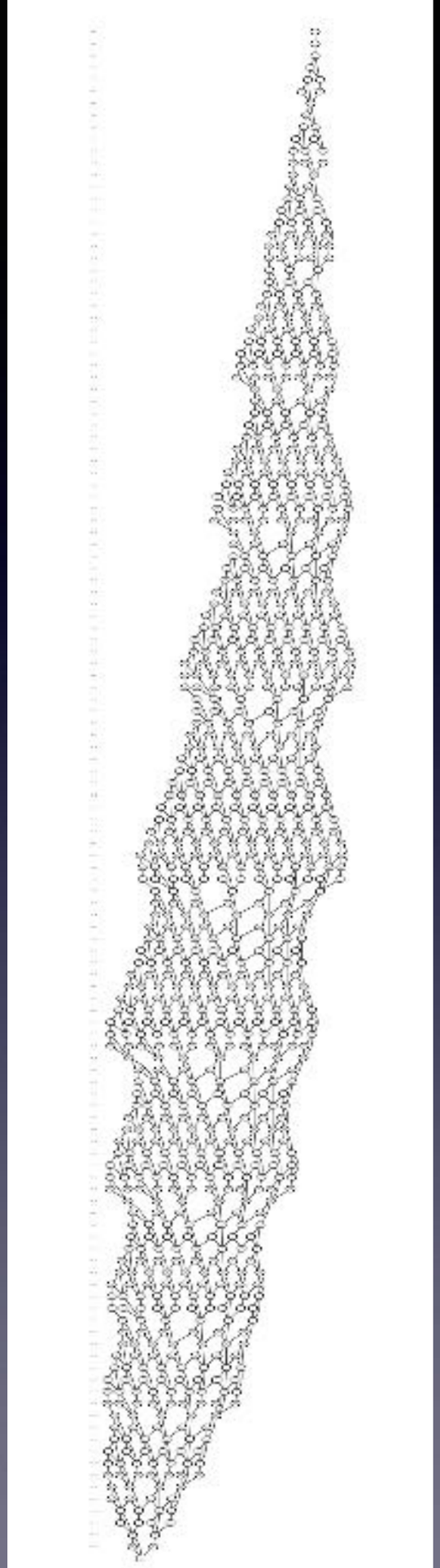
Column k



Column 2n-2



Only keep node with value N.
This becomes the bottom node of the BDD



BDD for multiplication of two 8-bit numbers

N=31439

891 nodes

$> 2^{110}$ paths

General properties of multiplication-BDD

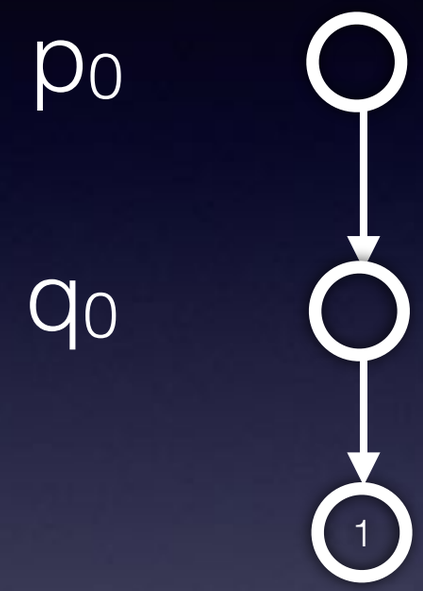
- BDD for n-bit numbers p and q has:
  - 2n different variables
  - $2n^2$ levels (each variable occurring n times)
  - $\leq 2n^3 - 9/2n + 5$ nodes

Using BDD for factoring

How to find values of p and q?

Finding lsb's of p and q

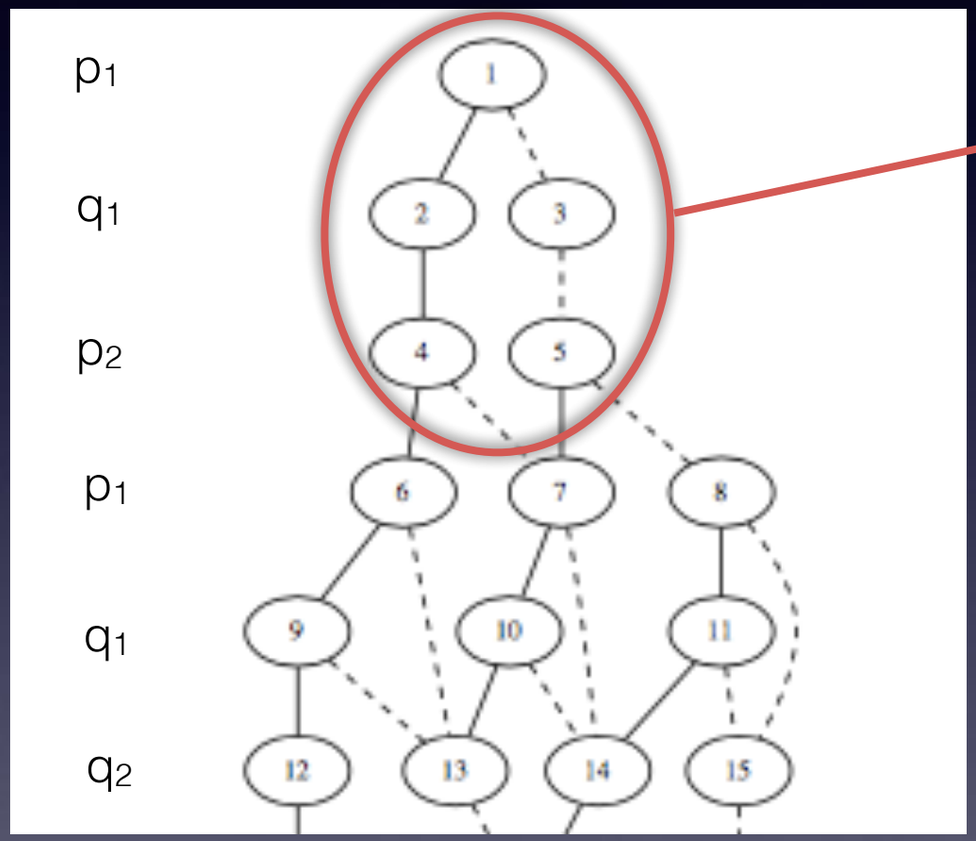
- When N is odd, top of BDD will always be



The two top levels give $p_0=q_0=1$

Fix $p_0=q_0=1$ in whole BDD by deleting 0-edges going out from levels with p_0 or q_0

Finding lsb's of p and q



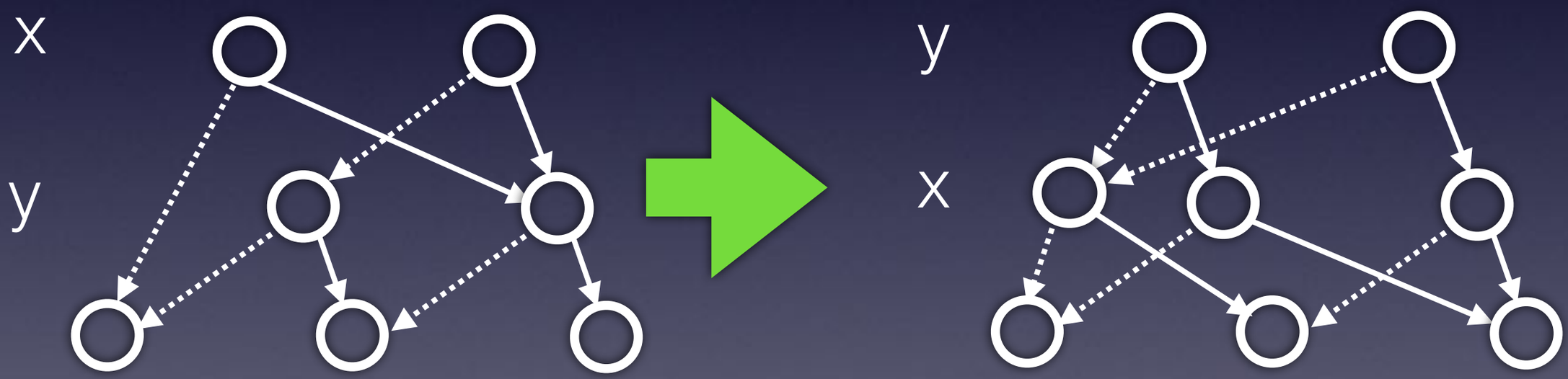
Gives $p_1=q_1$

Replace q_1 with p_1 on all levels

Depending on value of N, some more linear equations can be extracted from top of BDD

Swapping levels

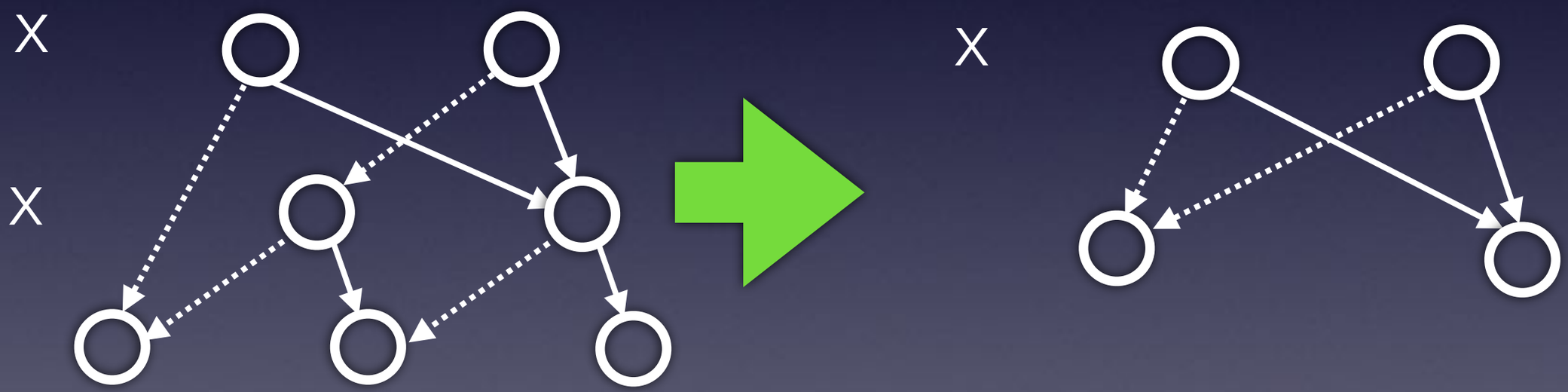
- There exists an algorithm for swapping adjacent levels



- Number of nodes may increase on affected level

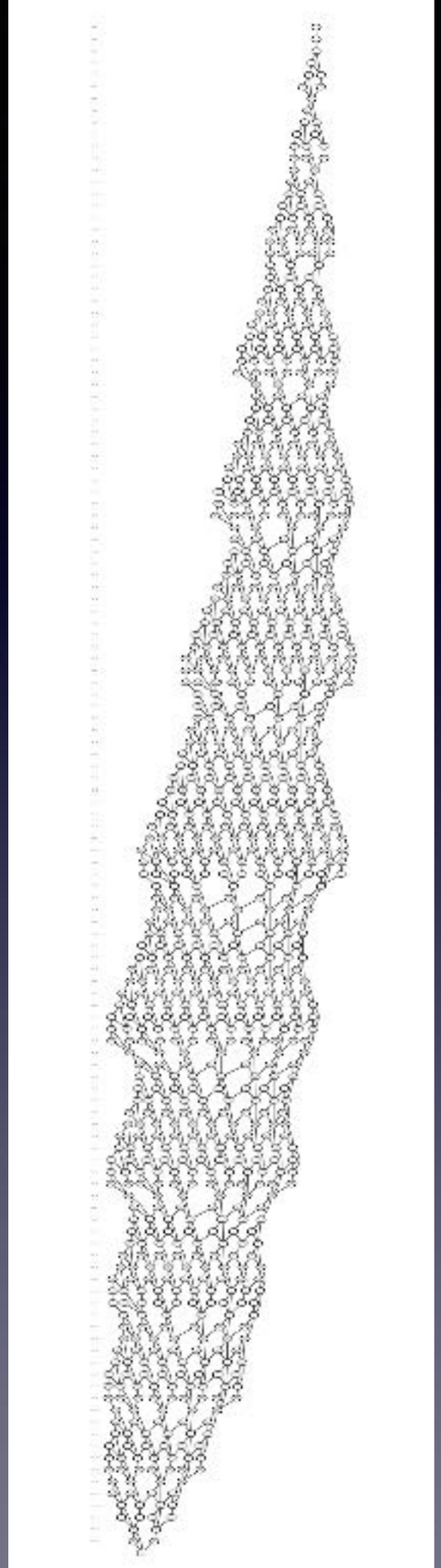
Absorbing dependency

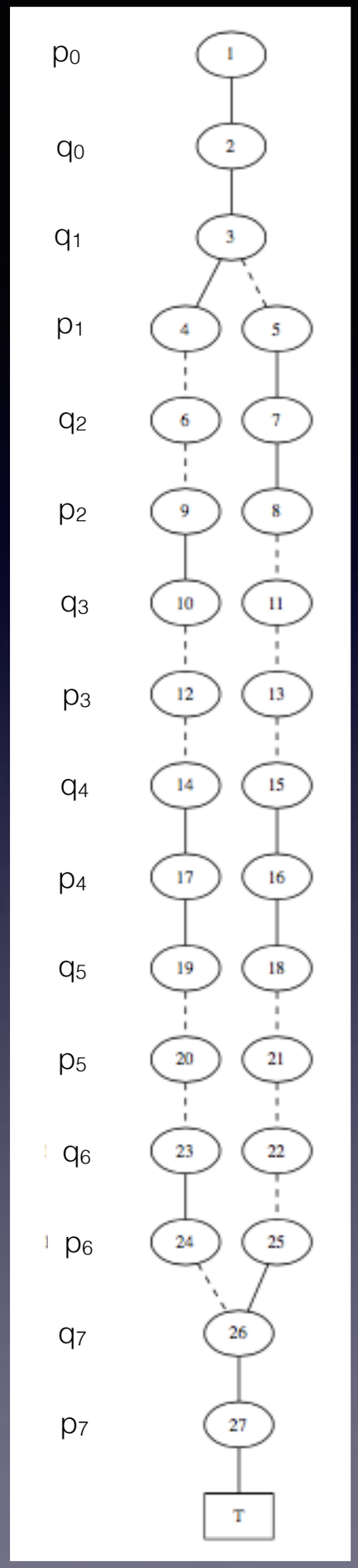
- Two adjacent levels with equal variables can be merged



Factoring algorithm

- Repeat until each p_i, q_j only occurs once in whole BDD:
  - Swap levels repeatedly such that two adjacent levels get equal variables
  - Merge levels
- Any path in resulting BDD gives values of p_i and q_j such that $pq=N$





Only two paths remain

p=211 and q=149

or

p=149 and q=211

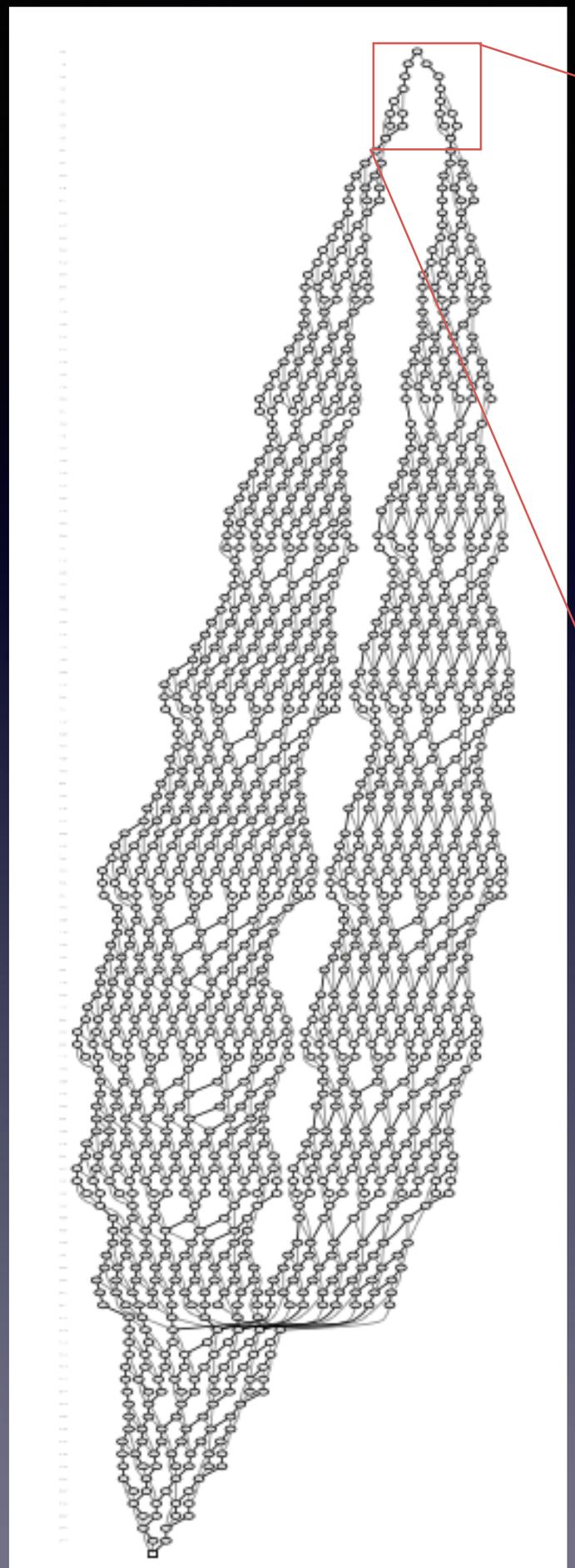
Number of nodes in BDD was never above 891 during solving

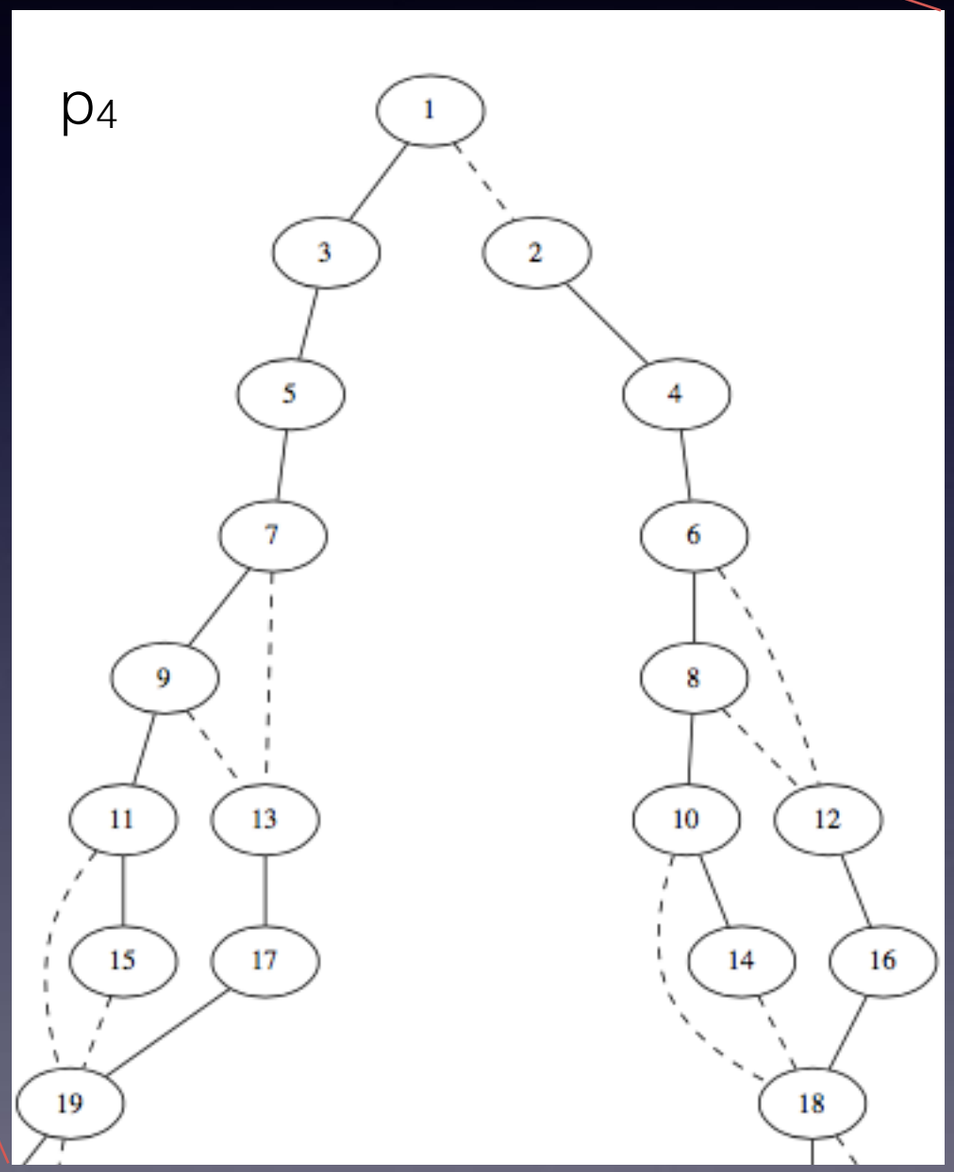
Factoring larger numbers?

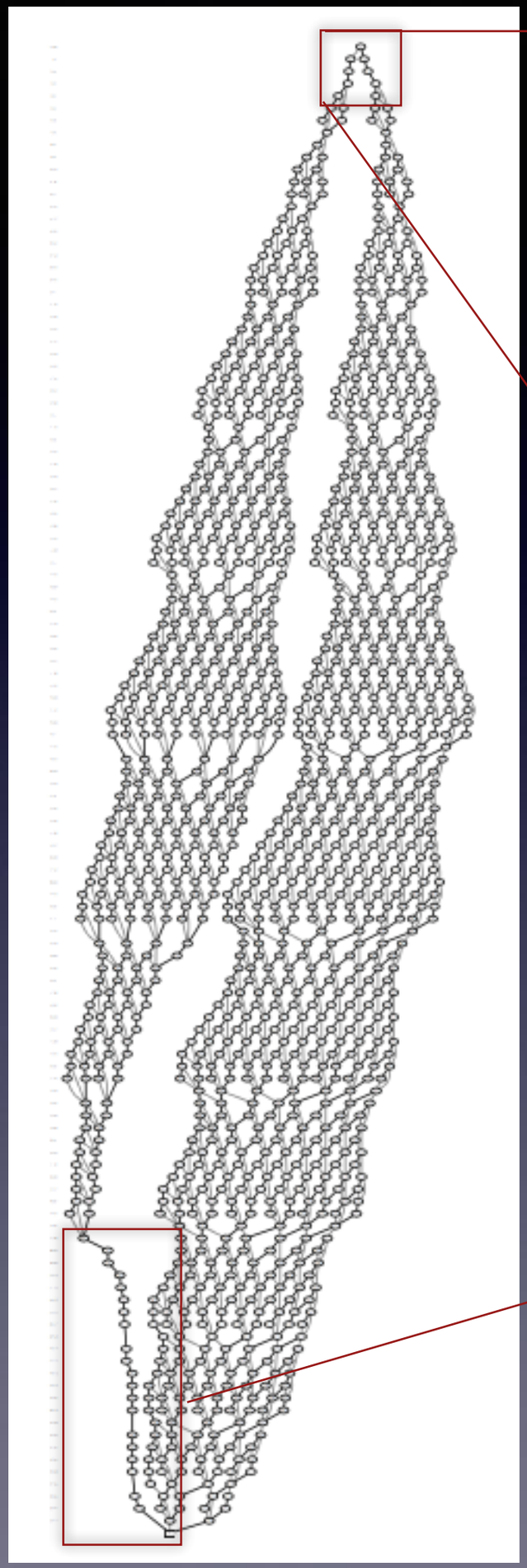
- Experiments show that simple swap-and-merge algorithm has complexity $\approx \sqrt{N} \approx 2^n$, measured as maximum number of nodes in BDD during solving

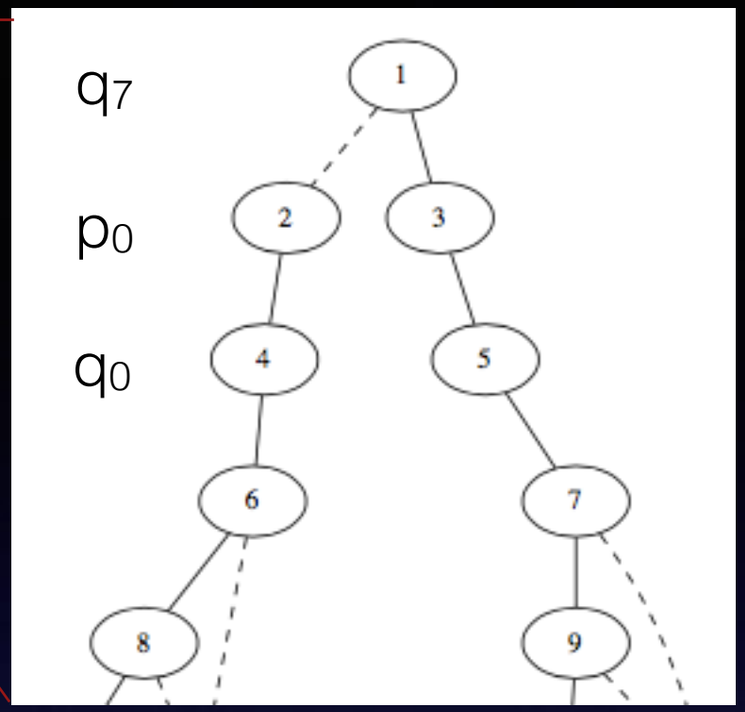
- However, BDD representation allows to try other tricks

All levels with p_4 merged and moved to the top









All levels with q_7 merged and moved to the top

$q_7=0$

⇓

$q_6=1$ $q_5=1$ $q_4=1$ $q_3=1$
$p_5=1$ $p_6=1$ $p_7=1$