Approximate-At-Most-k Encoding of SAT for Soft Constraints

Shunji Nishimura

National Institute of Technology, Oita College, Japan

PoS2023

At-most-k constraints and encodings

- the number of true values ≦ k
- problem: Boolean expressions will explode
- proposed encodings in the past:

  binary, sequential counter, commander, product, etc..

At-most-k constraints and encodings

- the number of true values $\leqq k$
- problem: Boolean expressions will explode
- proposed encodings in the past:

  binary, sequential counter, commander, product, etc..

are **absolutely** at-most-k

here is **approximately** at-most-k

At-most-k constraints and encodings

- the number of true values ≦ k
- problem: Boolean expressions will explode
- proposed encodings in the past:

  binary, sequential counter, commander, product, etc..

are **absolutely** at-most-k

covers all solutions

here is **approximately** at-most-k

only covers a part of solutions

Conventional vs Approximate

	solution coverage	purposes
conventional	complete	hard and soft constraints
approximate	incomplete	**only soft constraints**

- hard constraints: necessities
- solt constraints: to describe optional desires

Conventional vs Approximate

	solution coverage	purposes
conventional	complete	hard and soft constraints
approximate	incomplete	**only soft constraints**

but drastically reduces Boolean expressions

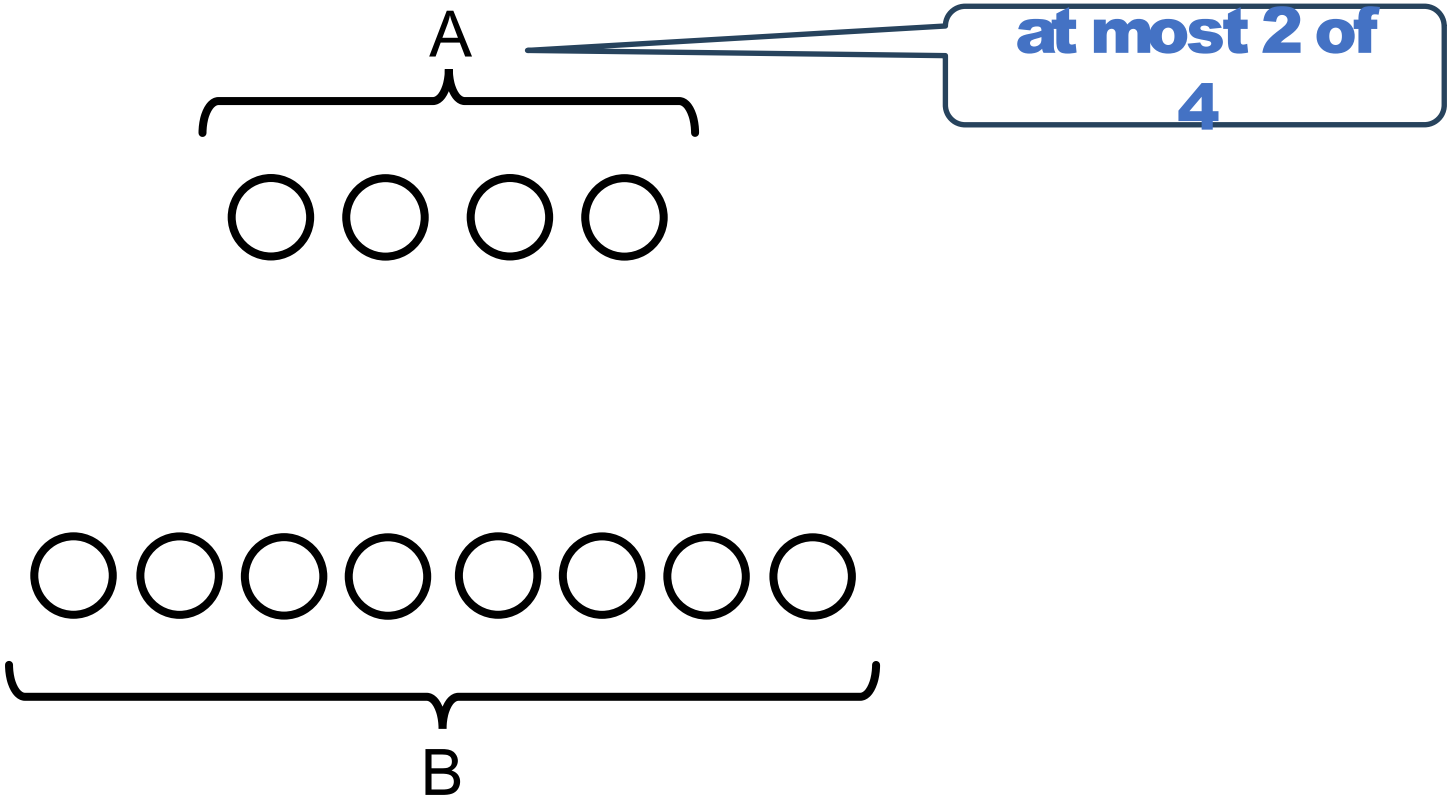
- hard constraints: necessities
- soft constraints: to describe optional desires

Soft constraints

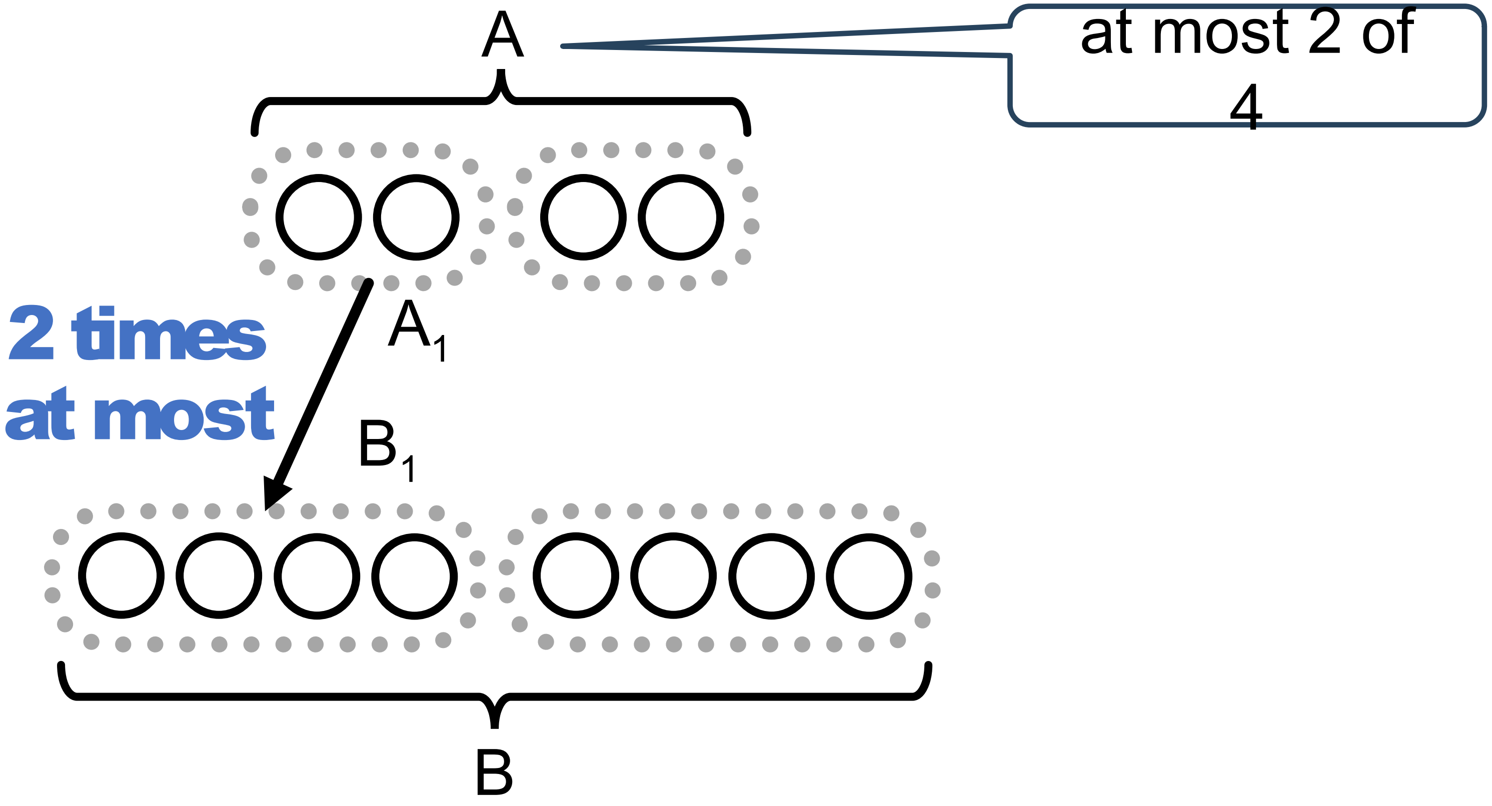
Not necessary but preferred

- In common with optimization problems
- Example: university timetabling
  - ➢ minimize empty time slots in between
  - ➢ minimize the number of teachers who have continuous classes
  - ➢ it is preferable a subject is always taught in the same room

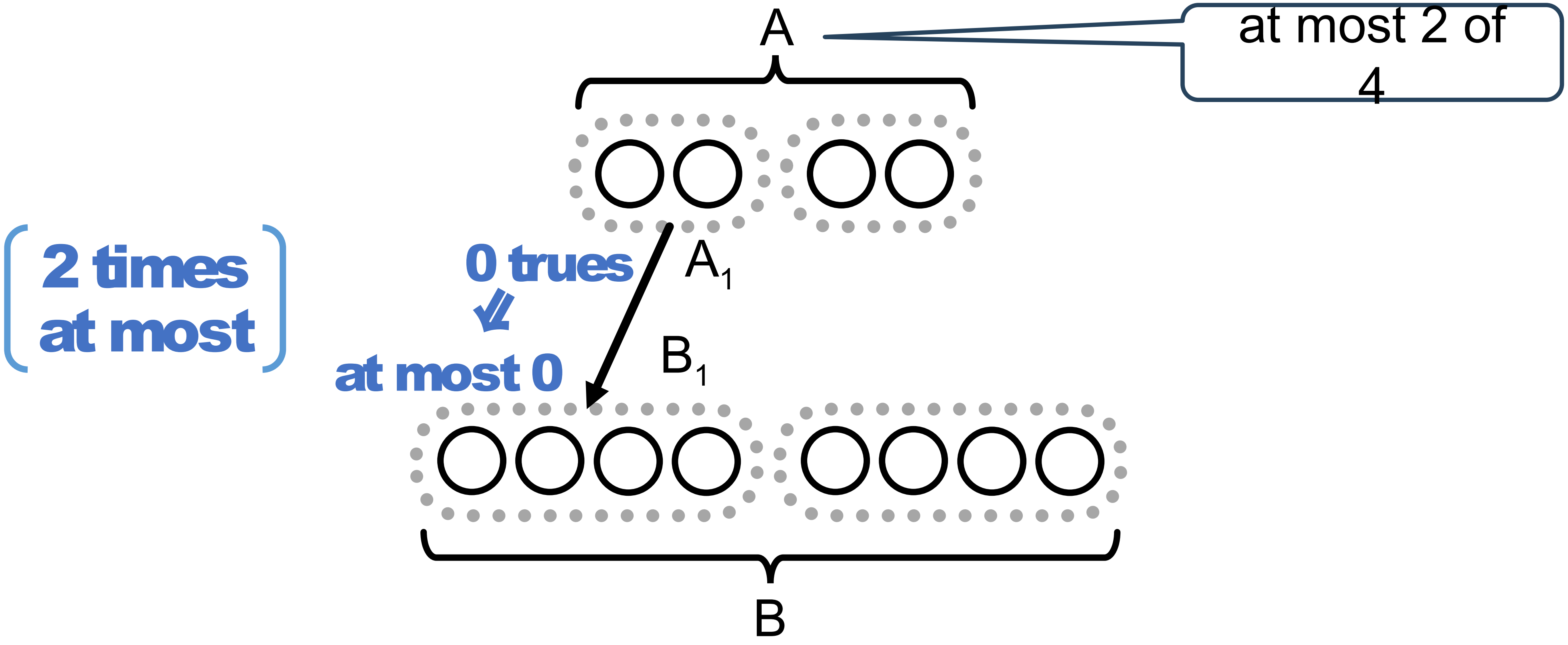
Fundamental idea



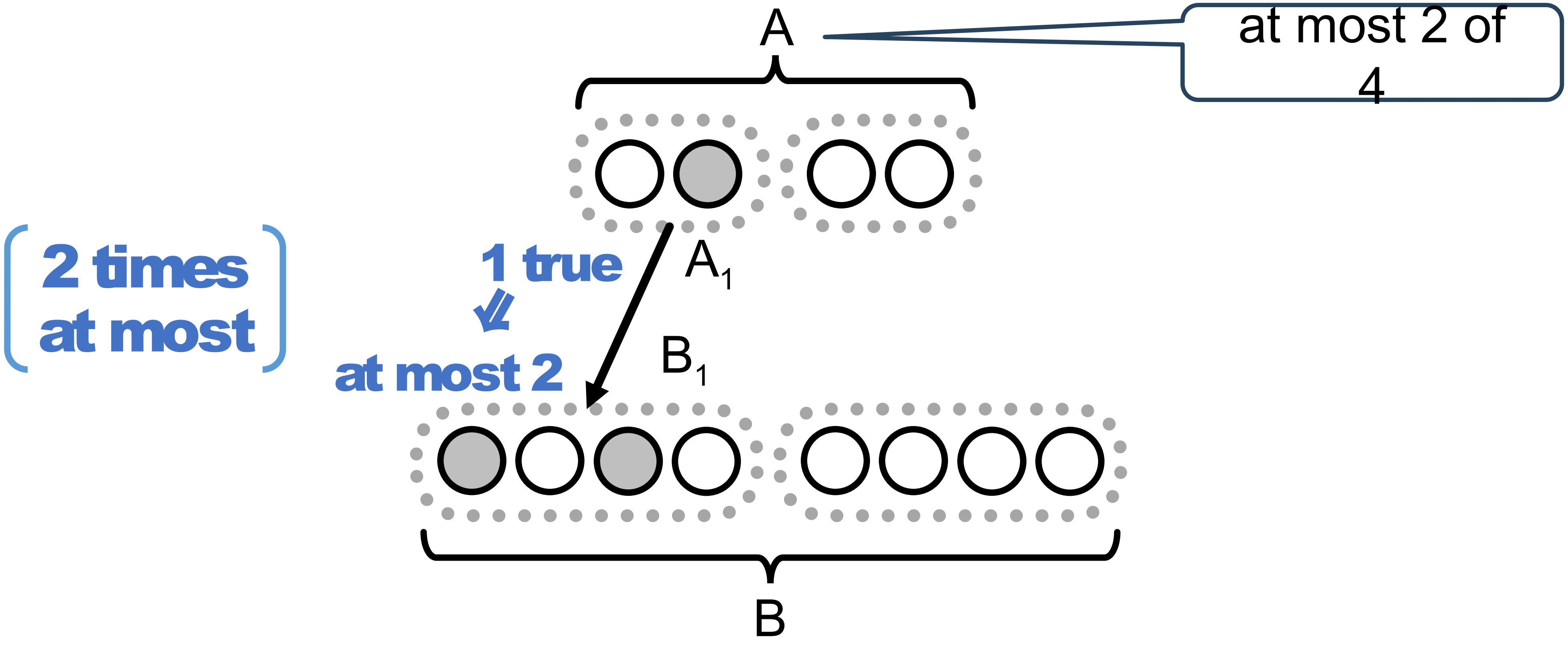
Fundamental idea



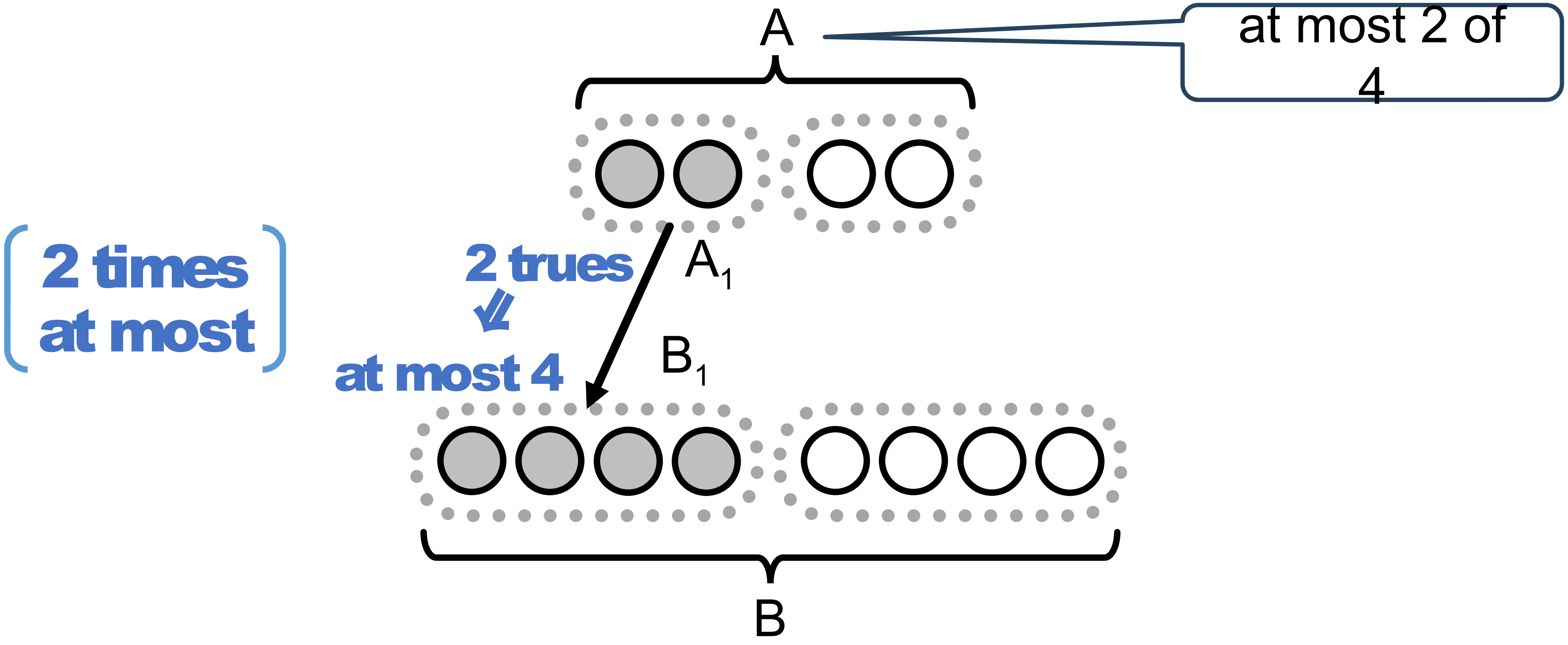
Fundamental idea



Fundamental idea



Fundamental idea



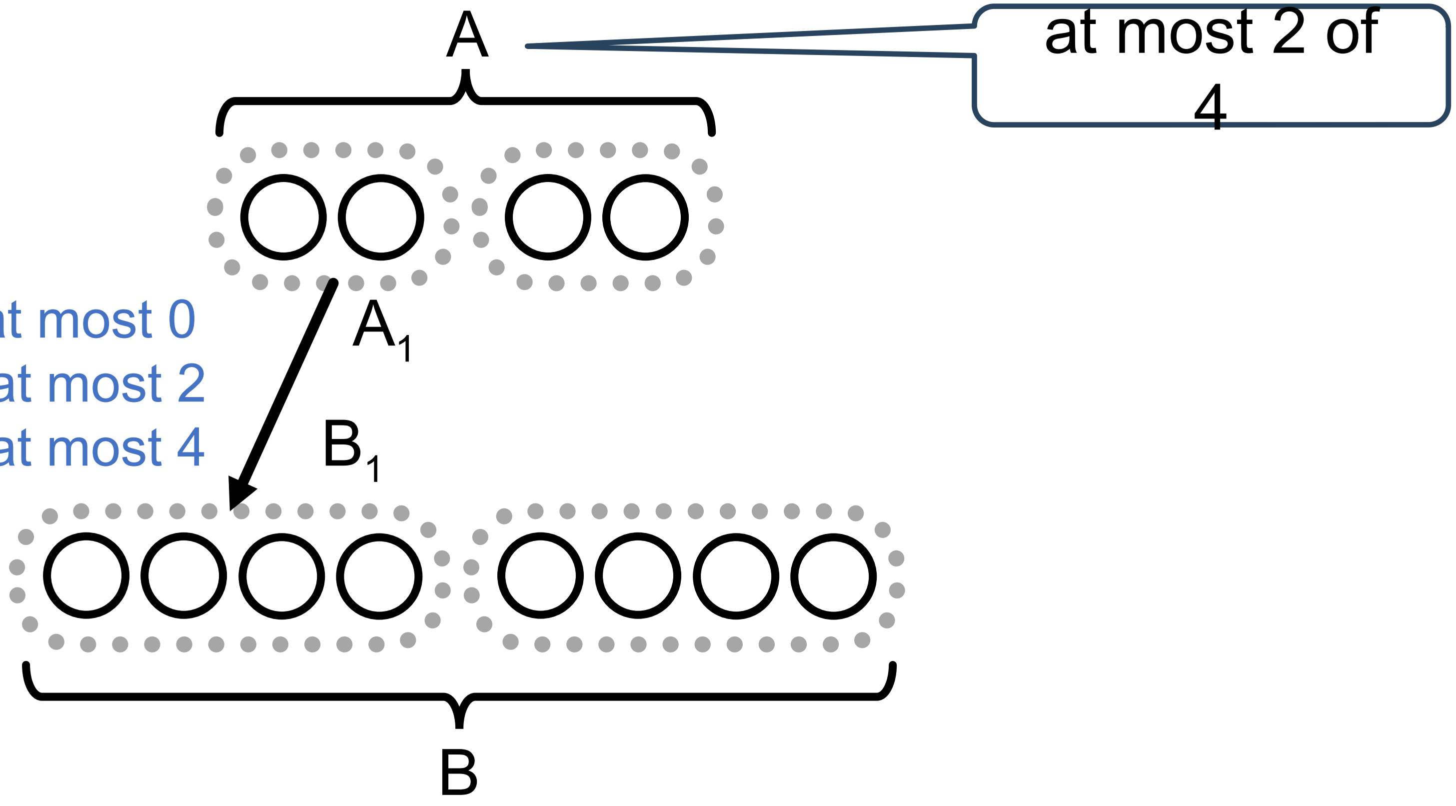
Fundamental idea

2 times at most =

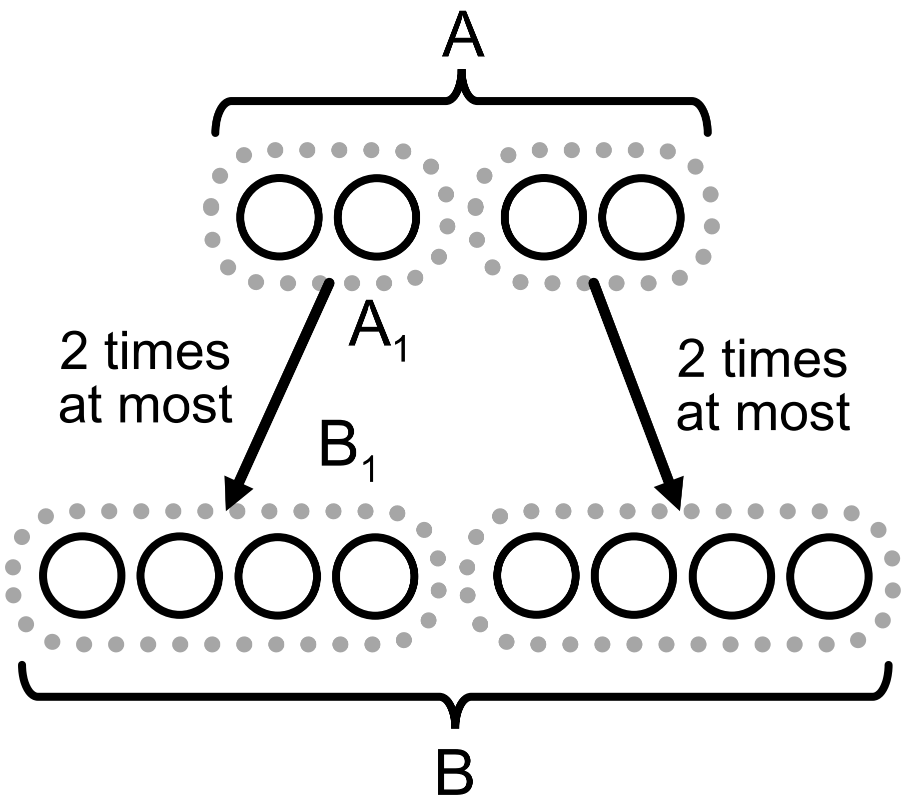
0 trues ⇒ at most 0

∧ 1 true ⇒ at most 2

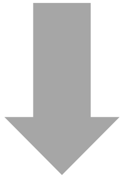
∧ 2 true ⇒ at most 4



Fundamental idea

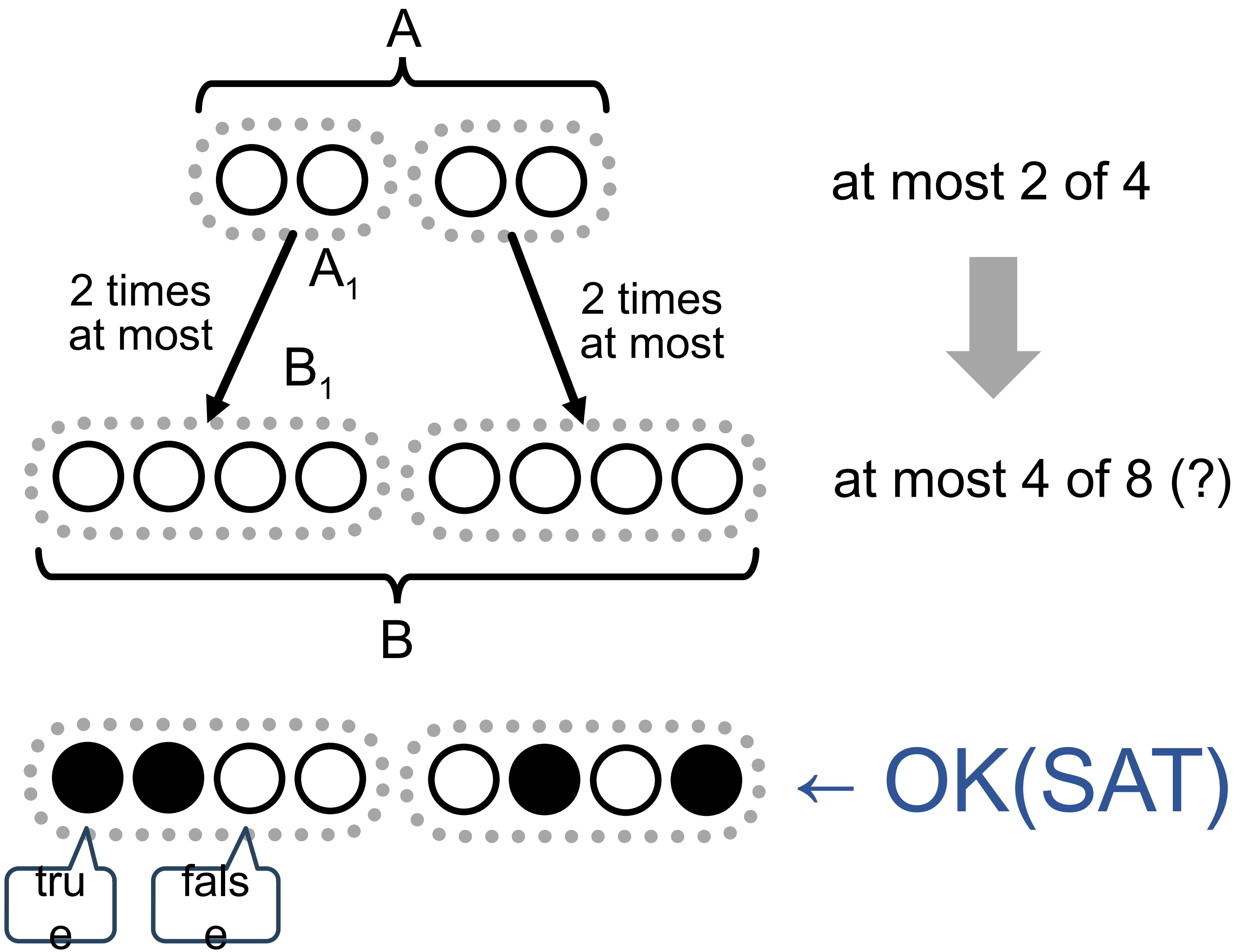


at most 2 of 4

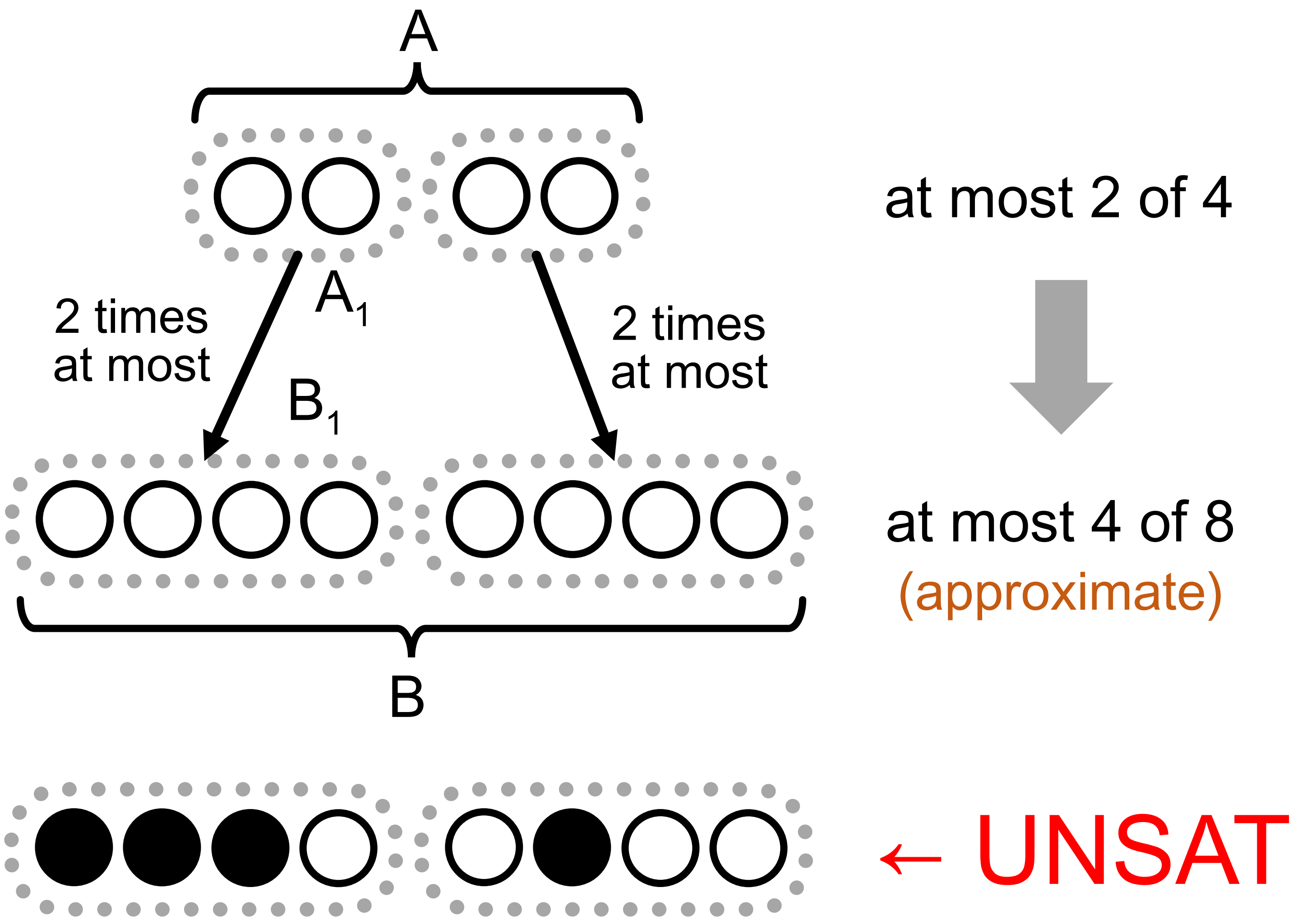


at most 4 of 8 (?)

Fundamental idea



Fundamental idea

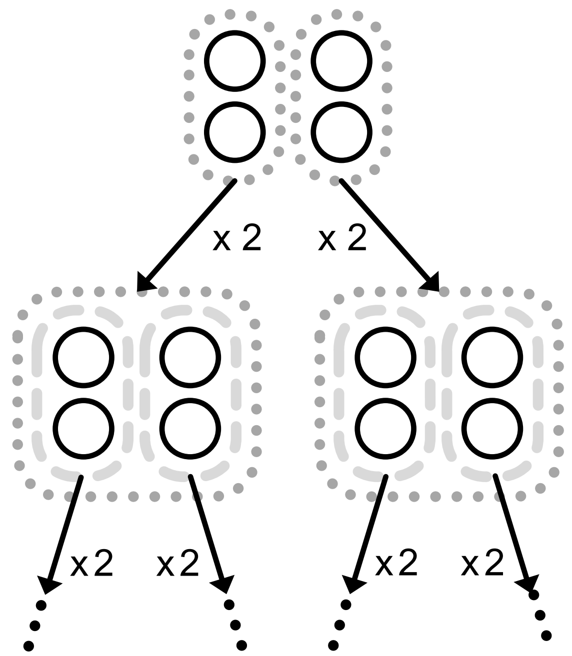


Approximate-At-Mo

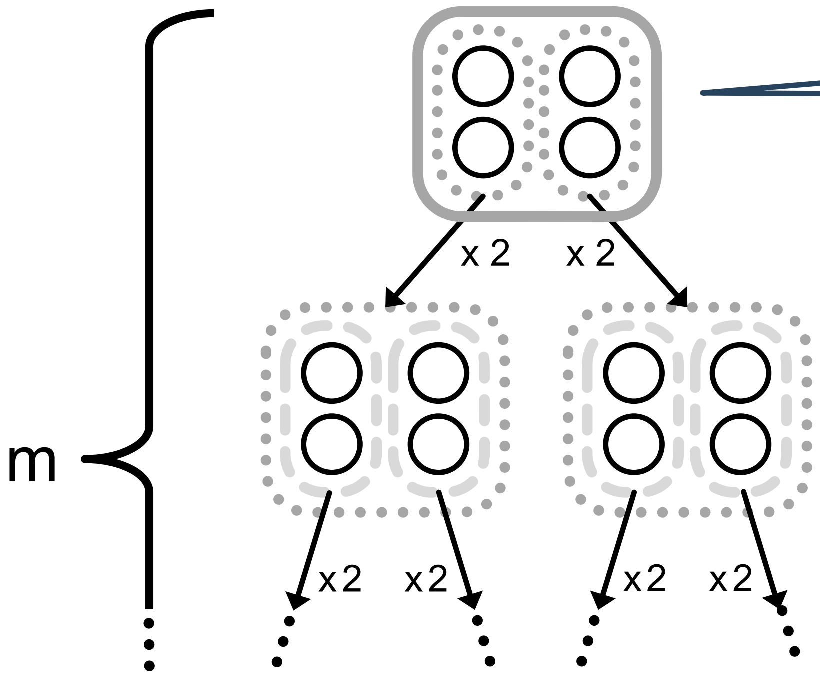
- **again, is not a real at-most-k**
- **should use for only soft constraints**

2x2 models

- two parents and four children
- define recursively



2x2 models

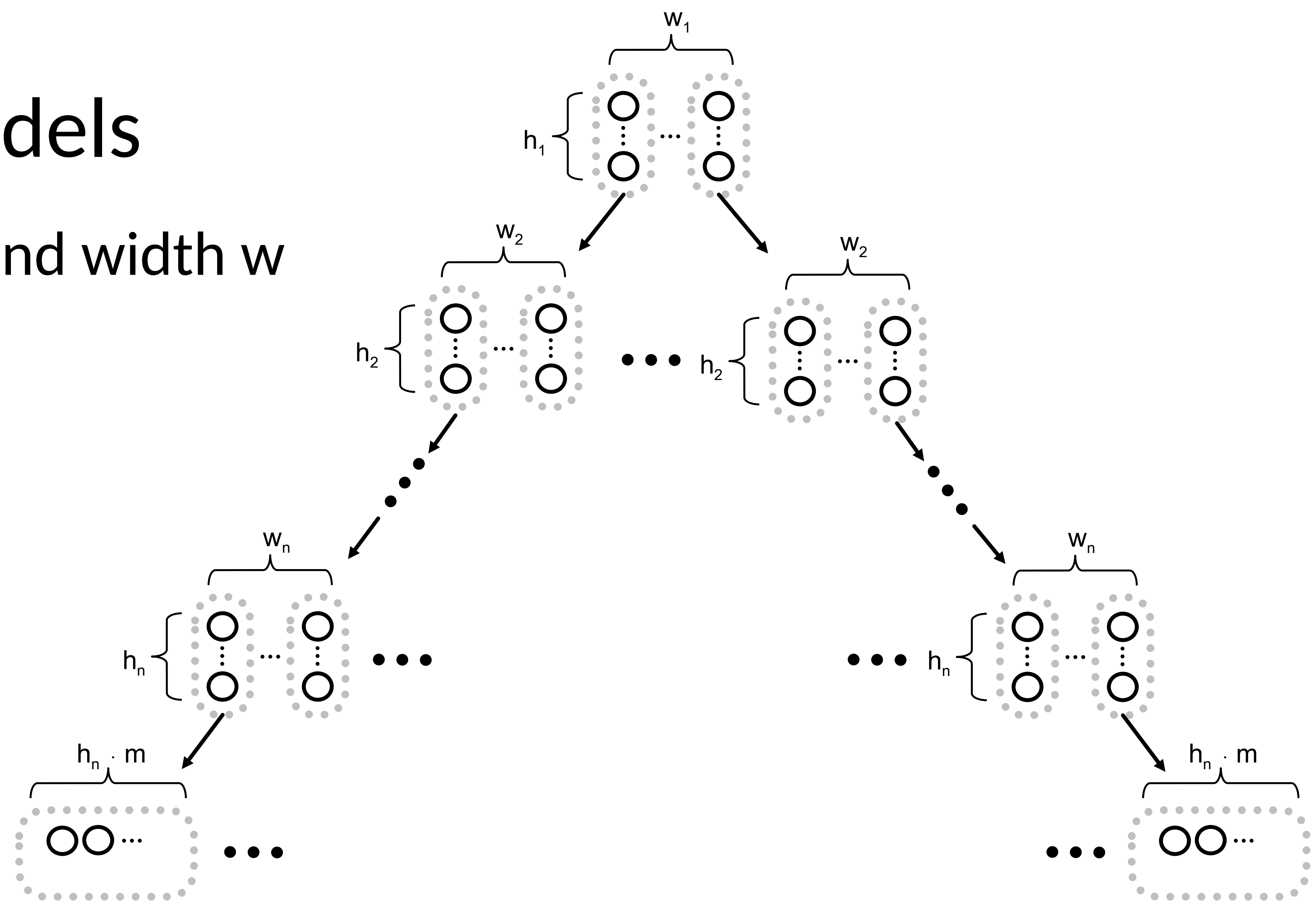


at most k of 4

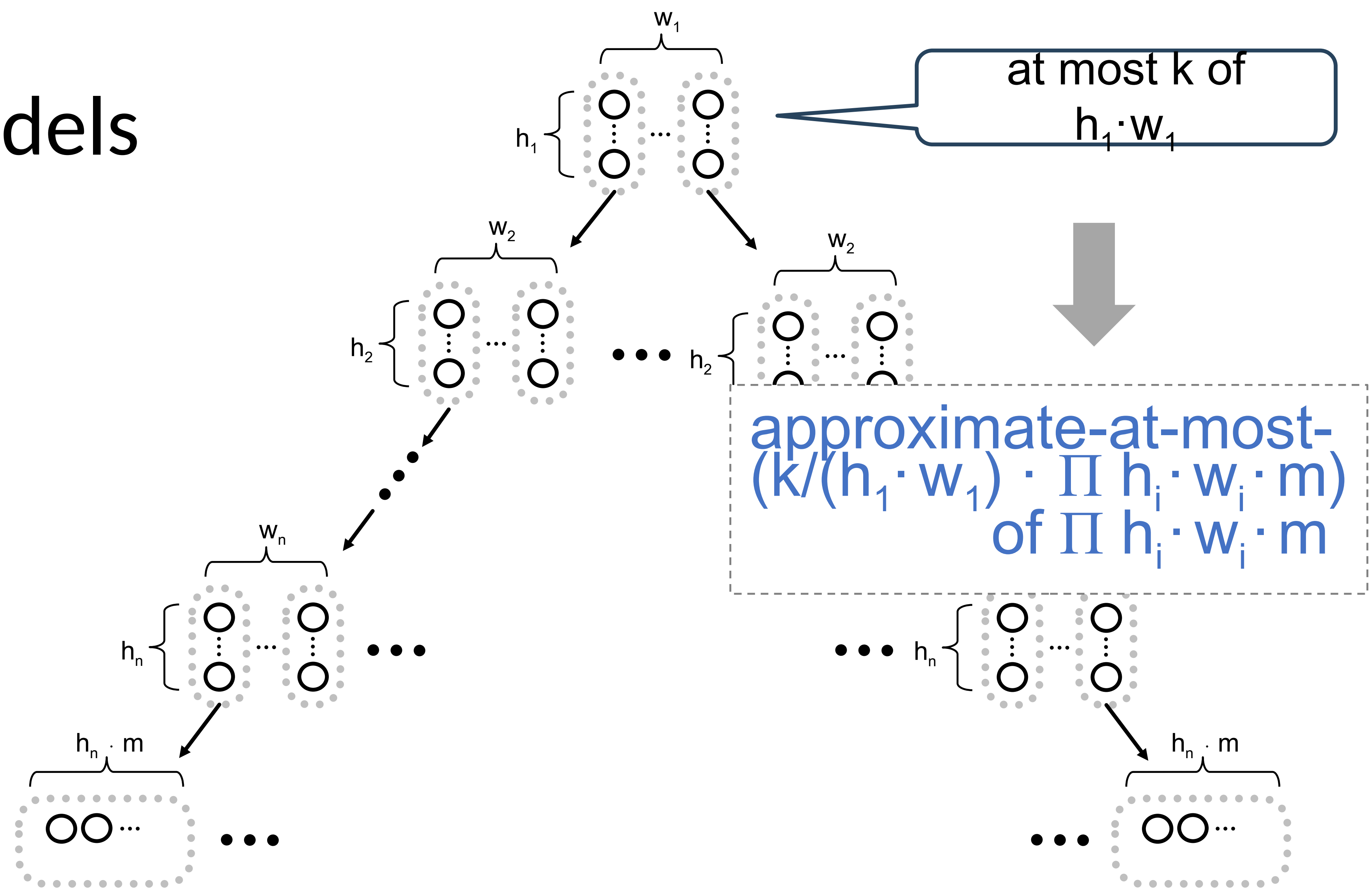
approximate-at-most-$(k/2 \cdot 2^{m+1})$ of 2^{m+1}

h x w models

- height h and width w

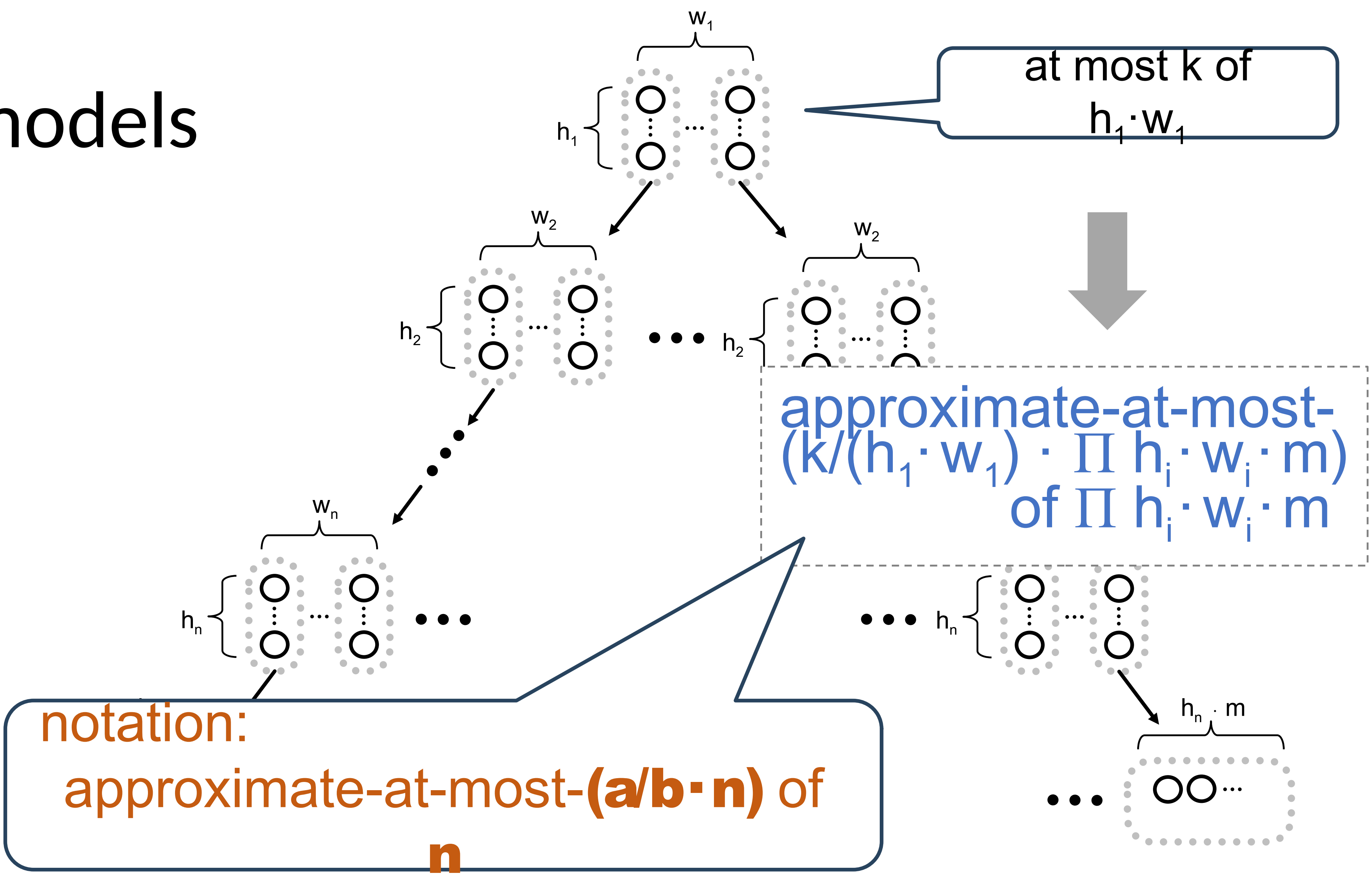


h x w models



w_1
h_1
at most k of $h_1 \cdot w_1$
w_2
h_2
w_2
h_2
approximate-at-most-$(k/(h_1 \cdot w_1) \cdot \prod h_i \cdot w_i \cdot m)$ of $\prod h_i \cdot w_i \cdot m$
w_n
h_n
h_n
$h_n \cdot m$
$h_n \cdot m$

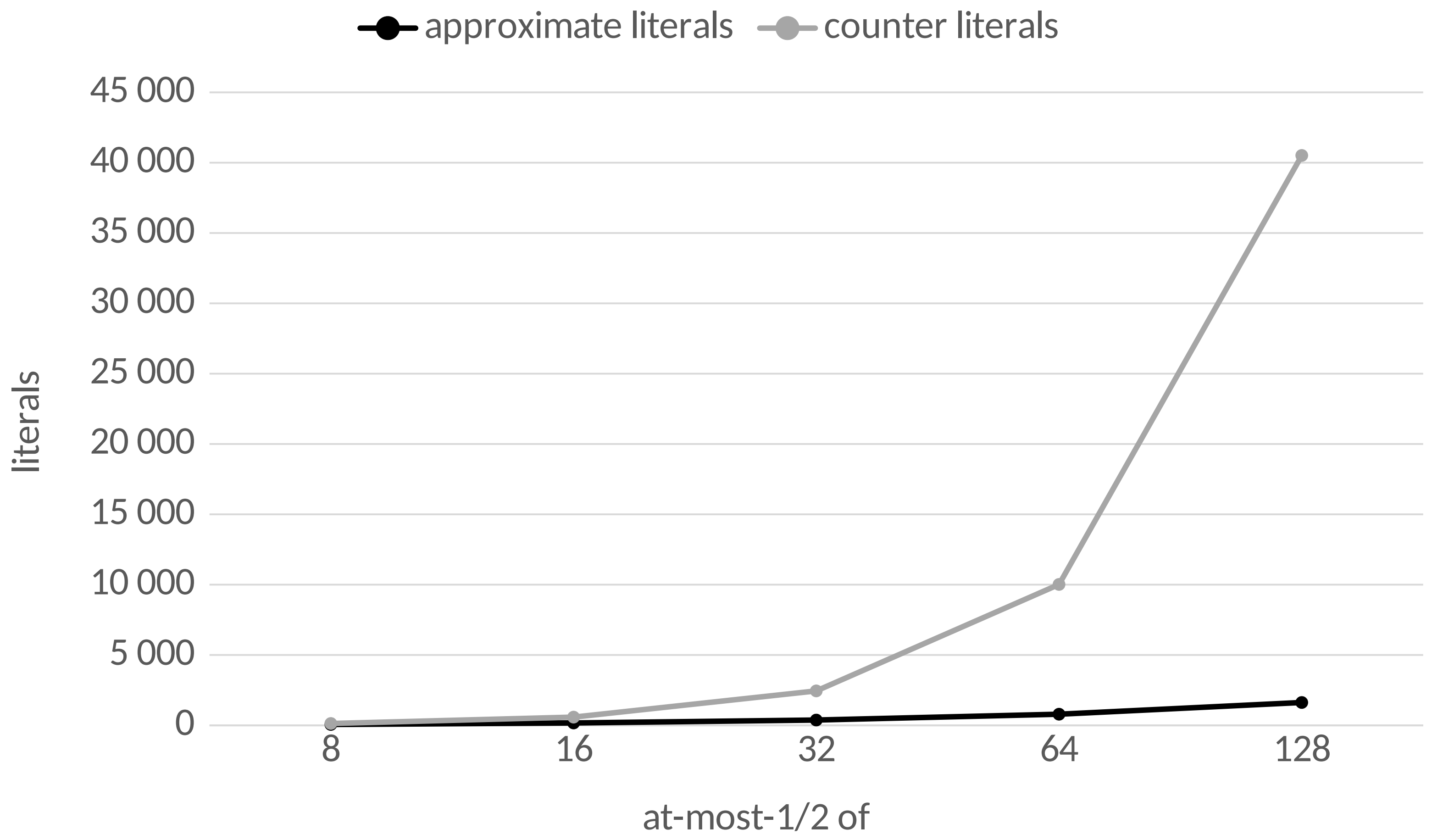
h x w models



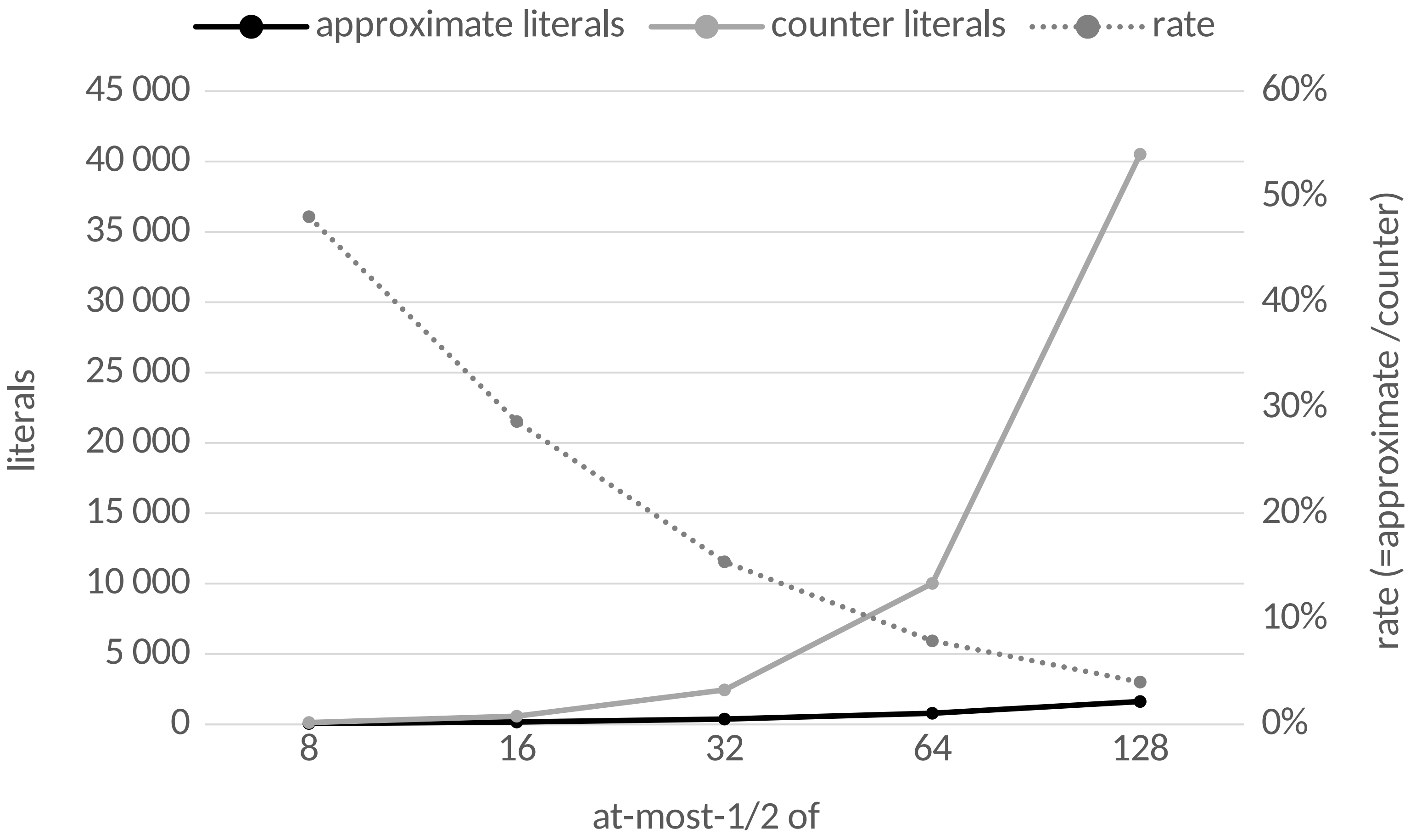
w1
h1
at most k of
h1·w1
w2
h2
w2
h2
approximate-at-most-
(k/(h1· w1) · Π hi· wi· m)
of Π hi· wi· m
wn
hn
hn
hn · m
notation:
approximate-at-most-(a/b·n) of
n

Literal number comparison (2x2 models)

vs sequential counter encoding

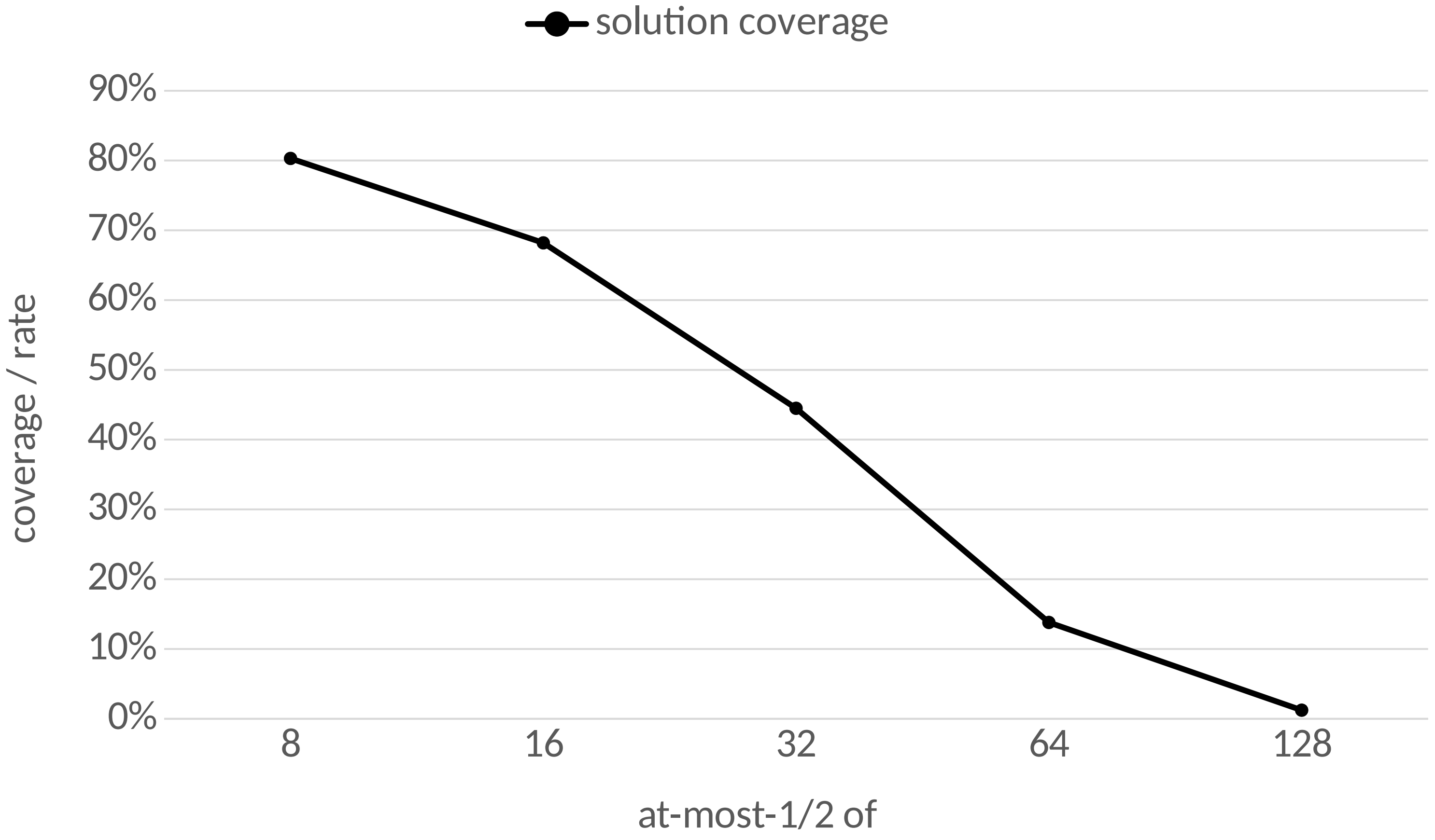


Literal number comparison (2x2 models)

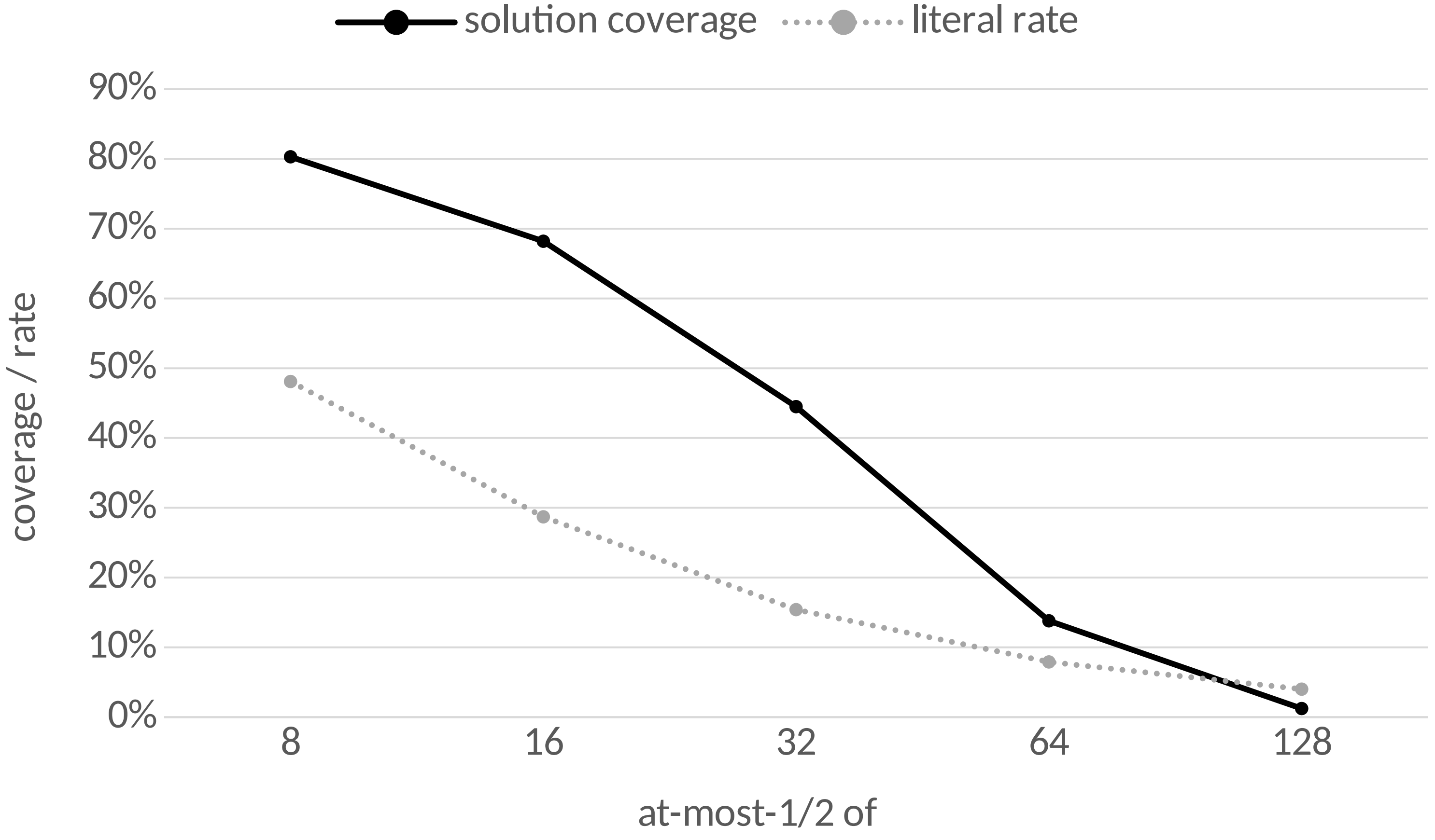


Coverages (2x2 models)

= (solutions by approximate) / (all solutions)

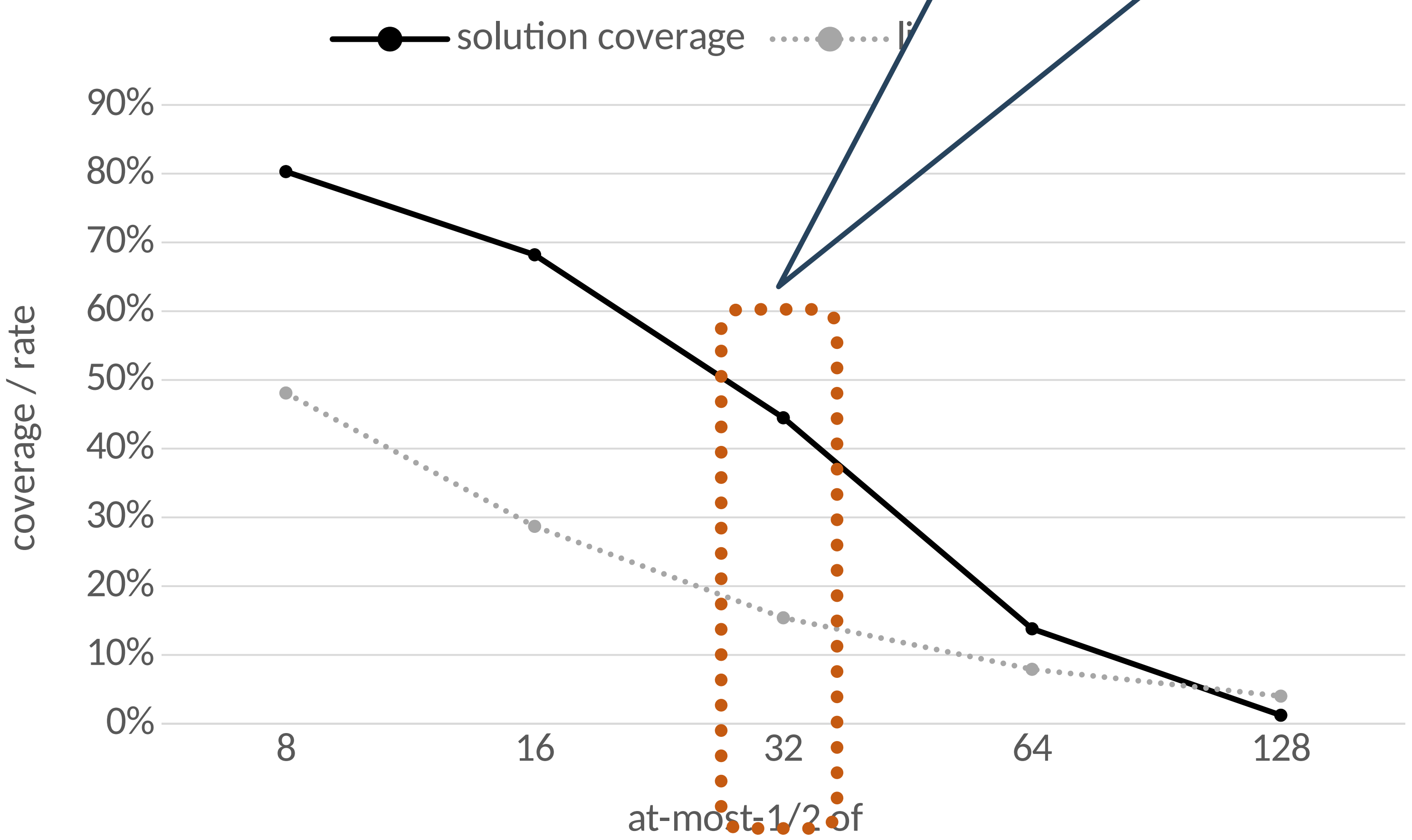


Coverages (2x2 models)

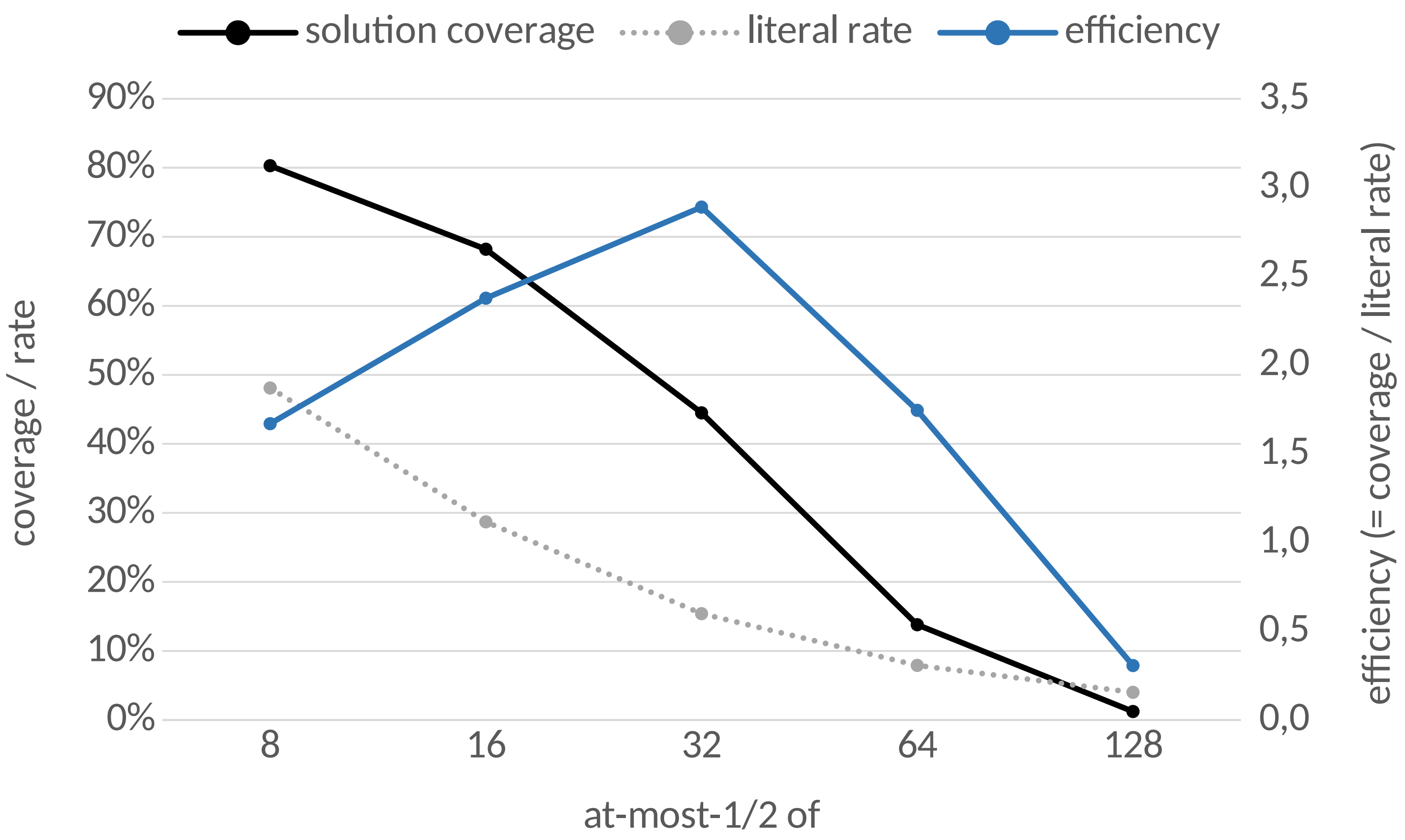


Coverages (2x2 models)

covers 44% on 15% literals



Coverages and efficiencies (2x2 models)



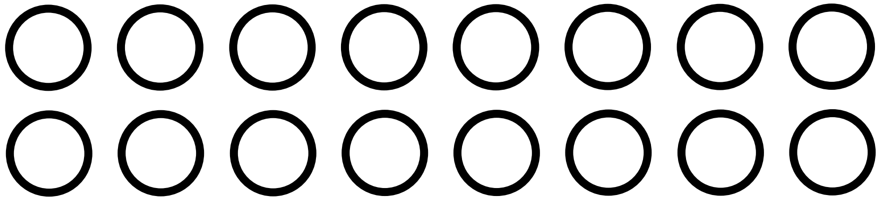
efficiency = coverage / literal rate

h x w models: adjustment

want to generate arbitrary **k of n**

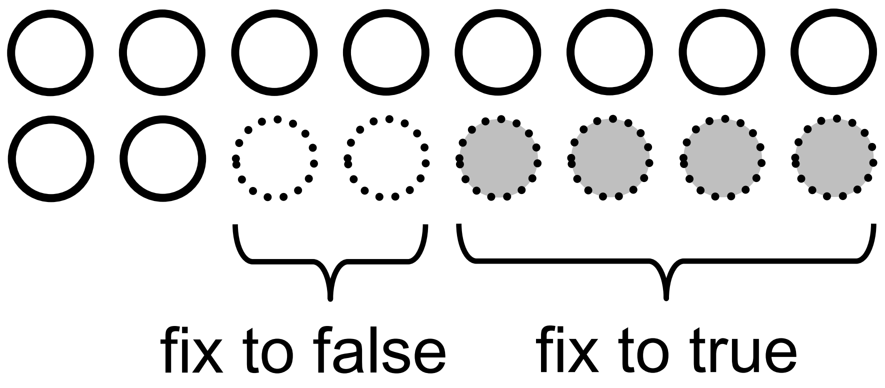
8 of 16

Target variables

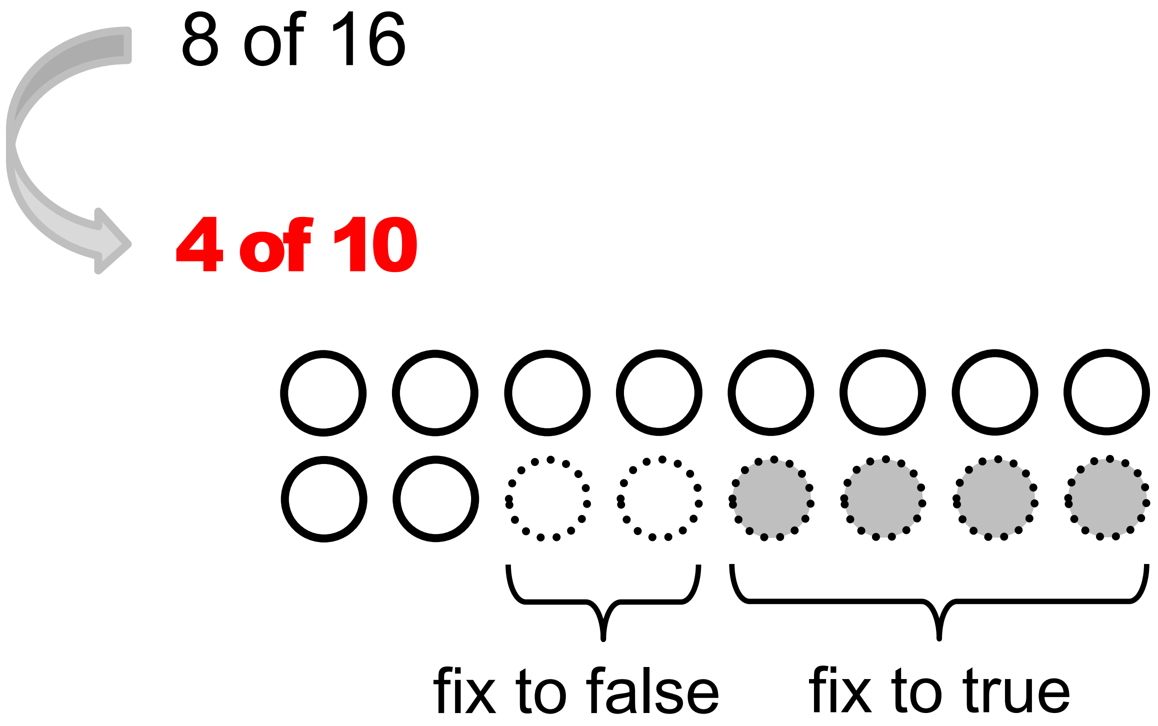


h x w models: adjustment

8 of 16



h x w models: adjustment

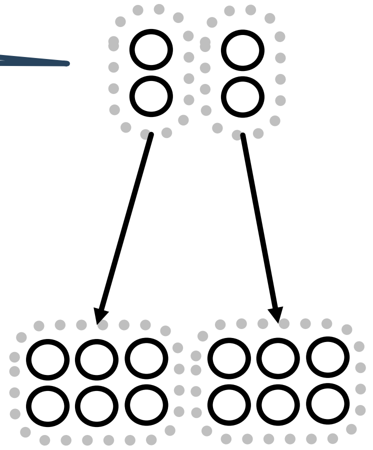


h x w models: example1

to generate **5 of 10**

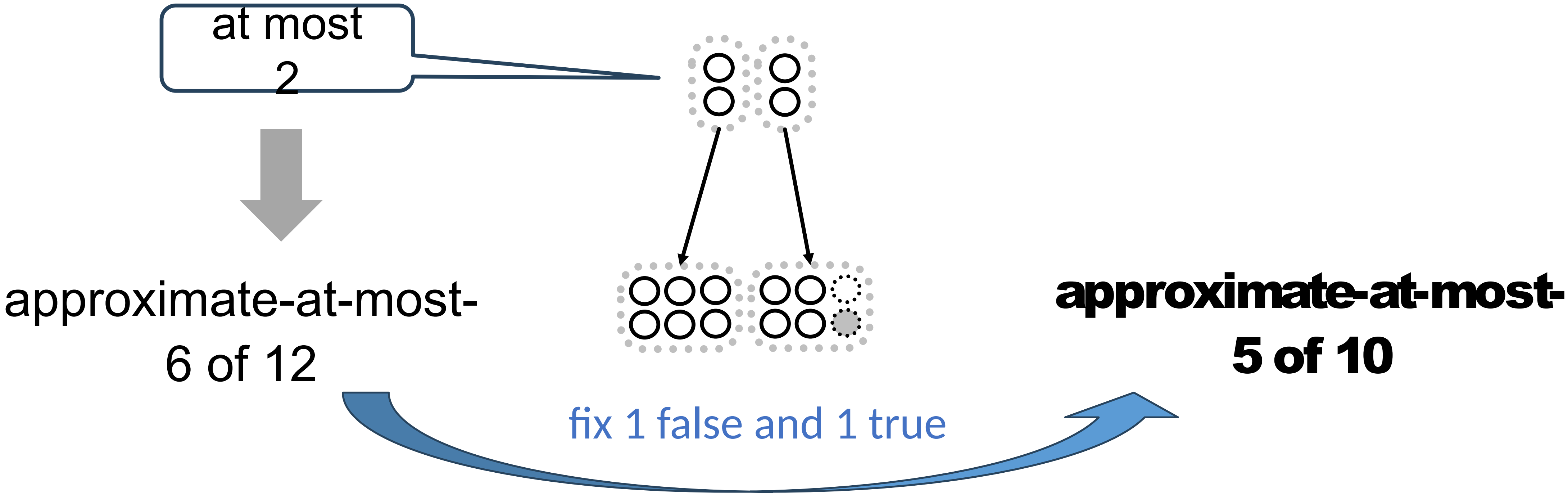
at most 2

approximate-at-most-6 of 12



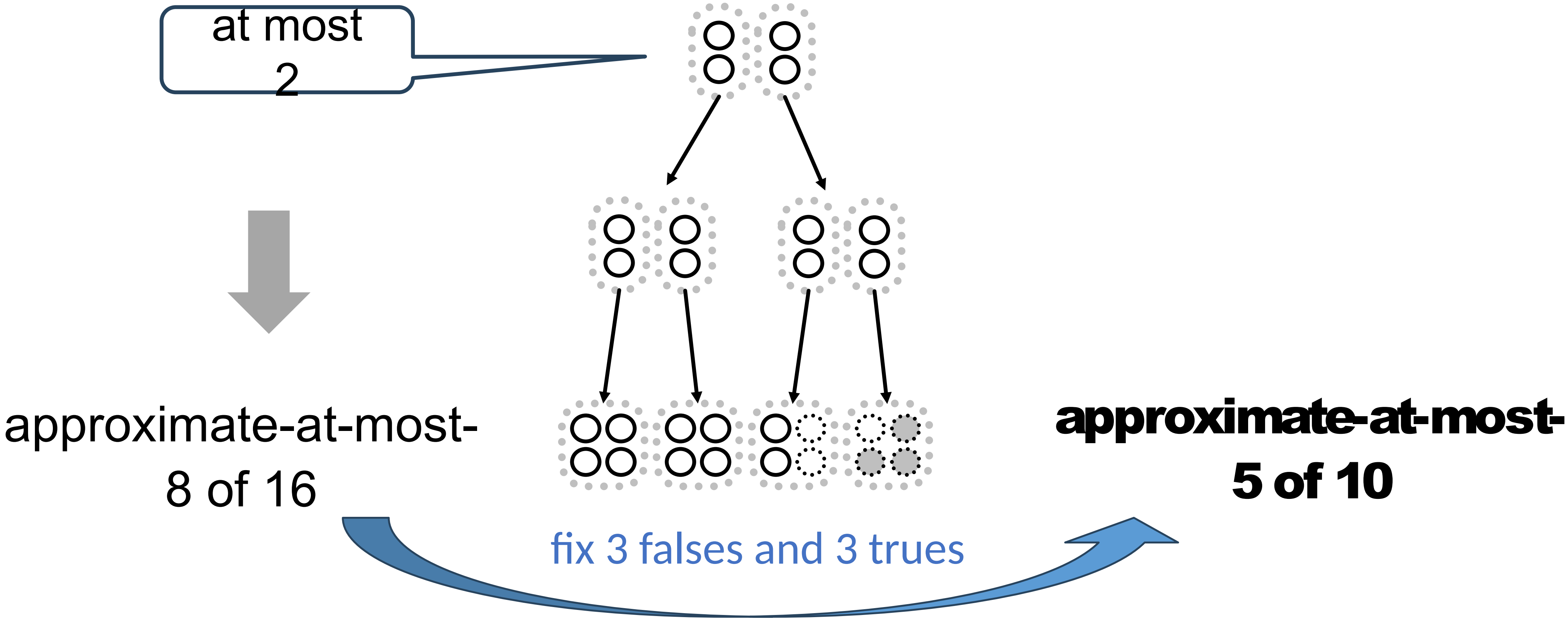
h x w models: example1

to generate **5 of 10**

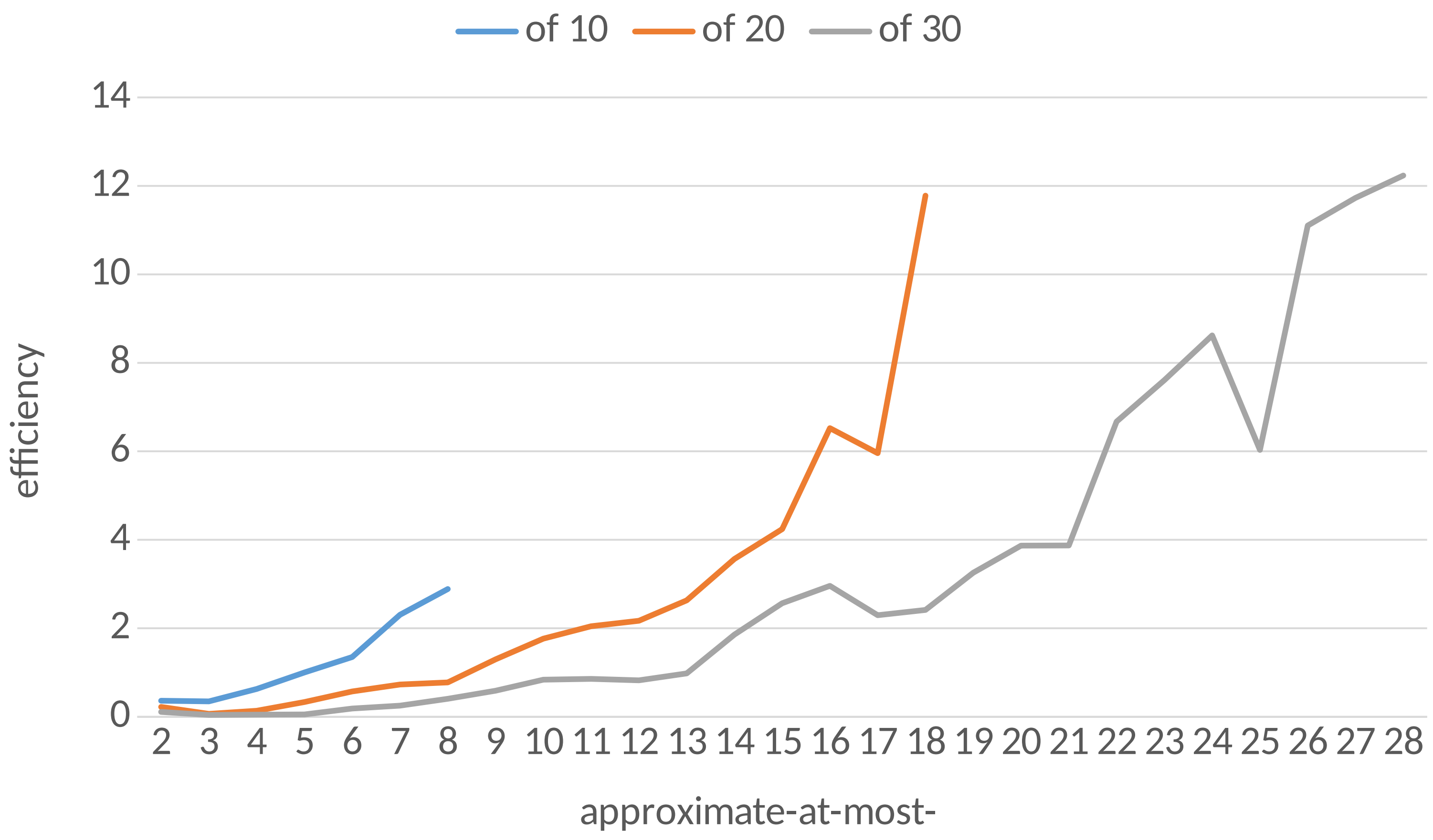


h x w models: example2

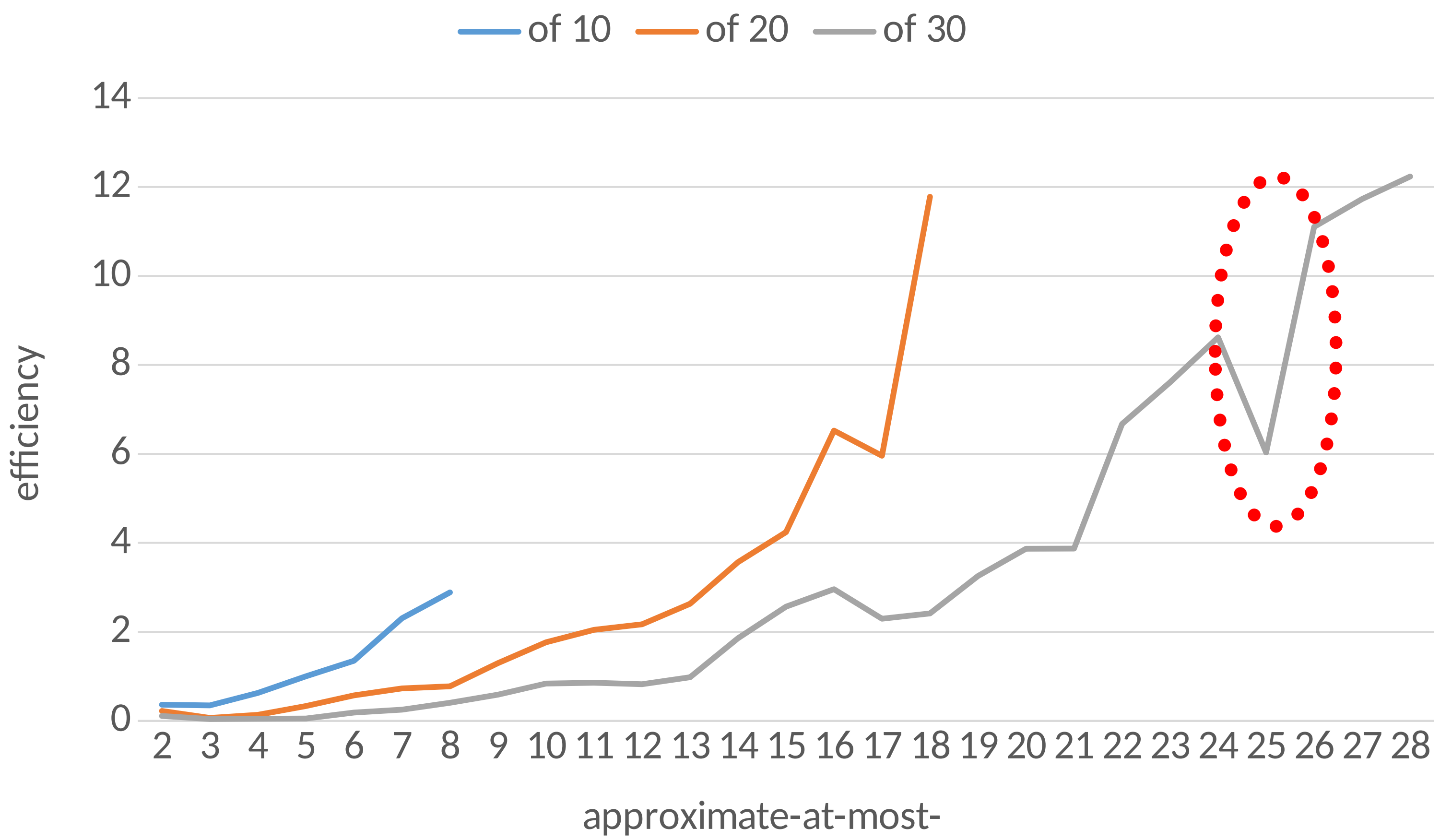
to generate **5 of 10**



The best efficiencies



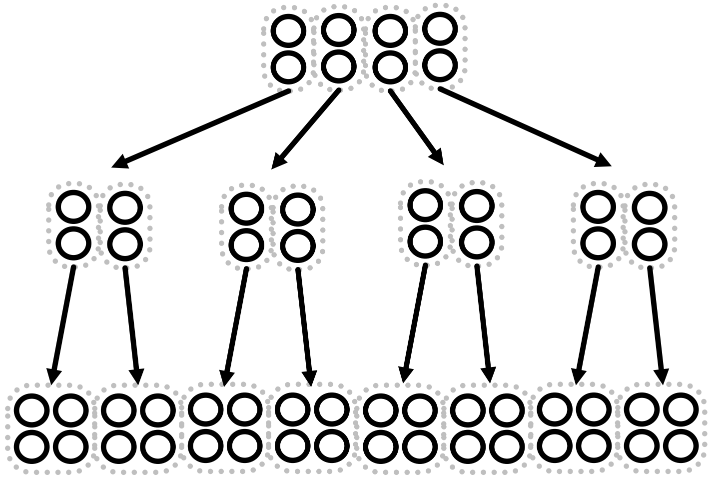
The best efficiencies



Low efficiency between highs

24 of 30 :
high efficiency

at-most-6

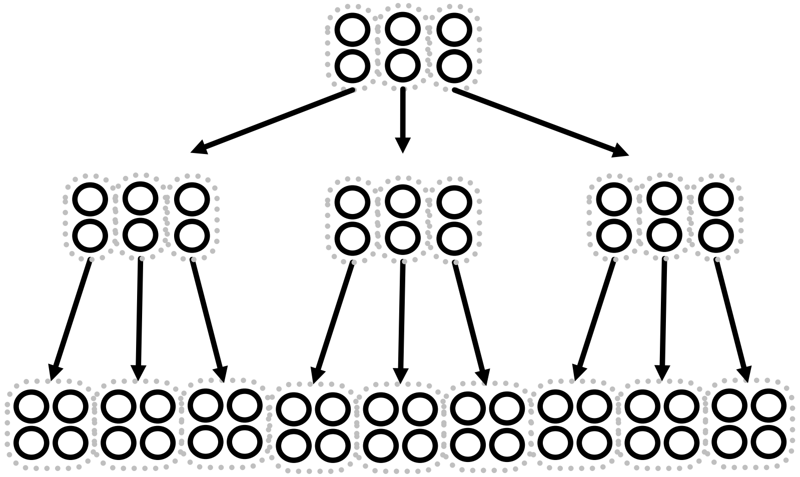


fix 2 falses and 0 trues

(24 of 32 → 24 of 30)

25 of 30 :
low efficiency

at-most-5

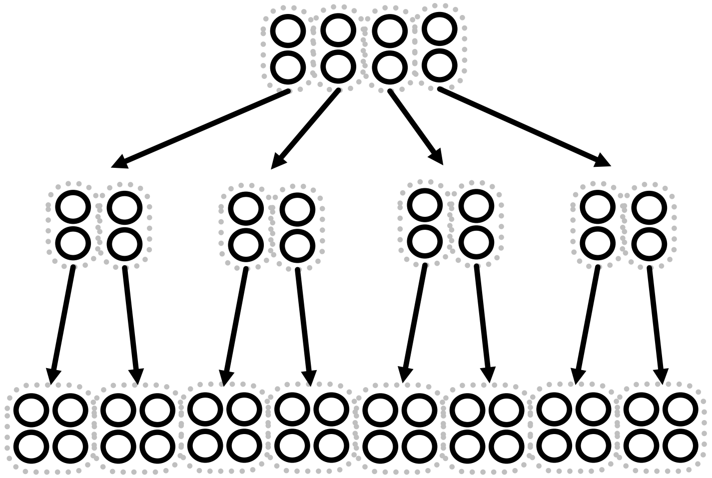


fix 1 false and 5 trues

(30 of 36 → 25 of 30)

26 of 30 :
high efficiency

at-most-7



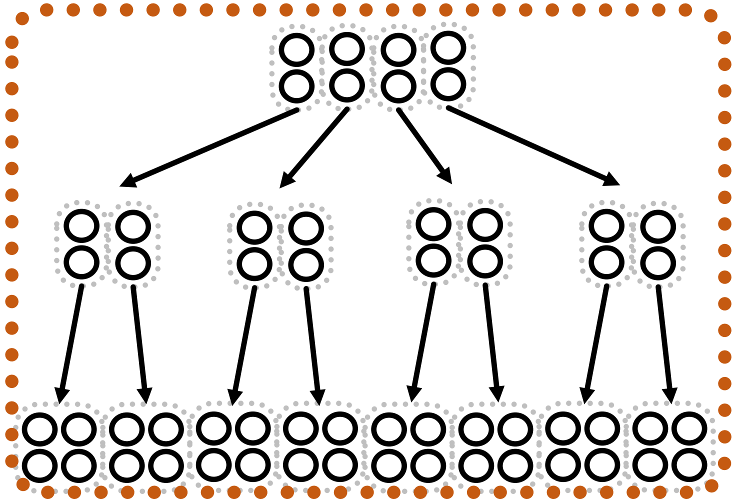
fix 0 falses and 2 trues

(28 of 32 → 26 of 30)

Low efficiency between highs

24 of 30 :
high efficiency

at-most-6

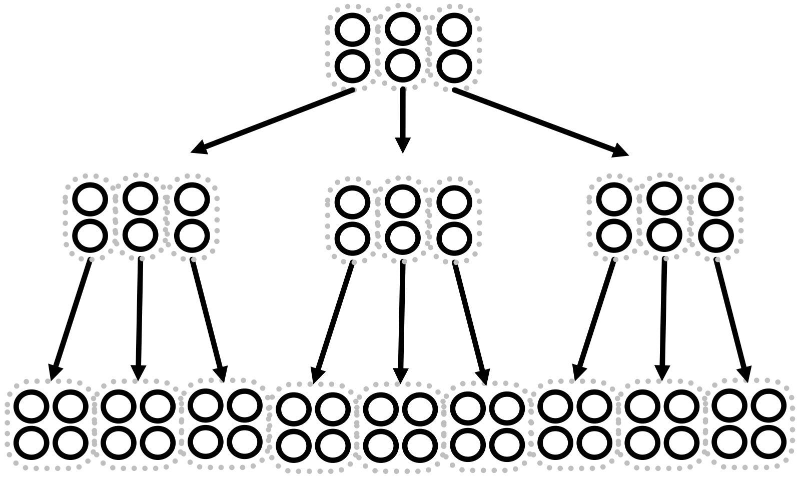


fix 2 falses and 0 trues

(24 of 32 → 24 of 30)

25 of 30 :
low efficiency

at-most-5

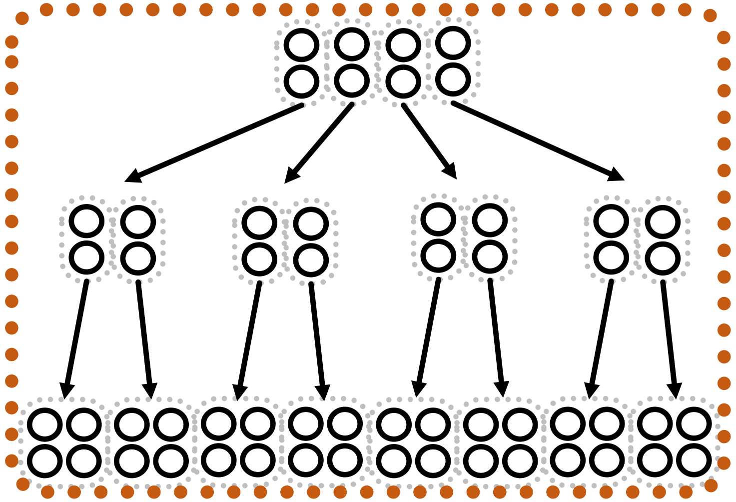


fix 1 false and 5 trues

(30 of 36 → 25 of 30)

26 of 30 :
high efficiency

at-most-7



fix 0 falses and 2 trues

(28 of 32 → 26 of 30)

Discussion1: coverage definition

all solutions

∪

at-most-8

∪

at-most-7

∪

:

∪

at-most-1

∪

no trues

Discussion1: coverage definition

all solutions

∪

at-most-8

∪

at-most-7

∪

:

∪

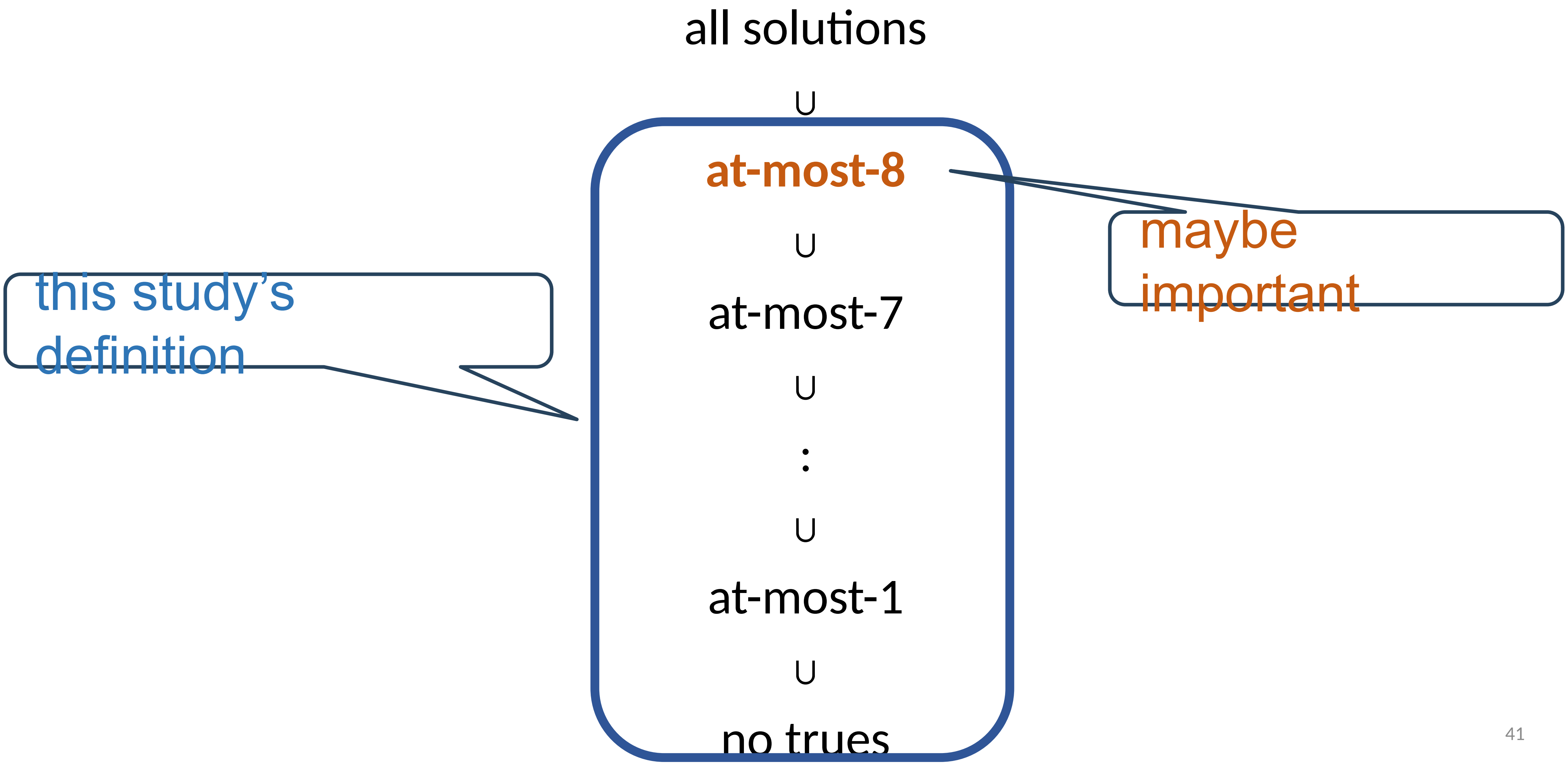
at-most-1

∪

no trues

this study's definition

Discussion1: coverage definition



Discussion2: probability of finding solutions

When approximate-at-most-k covers **50%** of the possible solutions, every single solution has probability **50%** to be found.

Discussion2: probability of finding solutions

When approximate-at-most-k covers **50%** of the possible solutions, every single solution has probability **50%** to be found.

↓

For a real-life problem ..

- has 1 solution → **50%** to find
- has 2 solutions → **75%** to find (whichever)

⋮

- has 10 solutions → **99.9%** to find (whichever)

⋮

Discussion2: probability of finding solutions

When approximate-at-most-k covers **50%** of the possible solutions, every single solution has probability **50%** to be found.

↓

For a real-life problem ..

- has 1 solution → **50%** to find
- has 2 solutions → **75%** to find (whichever)

  ⋮

- has 10 solutions → **99.9%** to find (whichever)

  ⋮

coverage ≠ finding probability

Conclusion

- at-most-k constraints are recursively applied (with multiplying)
- less Boolean expressions needed than conventional encodings,

  but does not cover all solutions
- available for searching better solutions under soft constraints
- Ex. at-most-16 of 32
  - ➢ only 15% of literal number (vs sequential counter)
  - ➢ covers 44% of the solution space