Analyzing and Avoiding Pathological Behavior in Parallel Best-First Search

Ryo Kuroiwa (University of Toronto)
Alex Fukunaga (University of Tokyo)

Parallel Best-First Search (BFS)

- BFS is used to solve graph search problems such as planning
- Many parallelization methods have been previously proposed
- Previous parallel A* methods have been experimentally shown to scale well

Method	BFS	Environment	Citation
HDA*	A*	distributed	Kishimoto et al. 2009
PBNF	A*, WA*	multi-core	Burns et al. 2010
KPBFS	WA*	multi-core	Vidal et al. 2010
HDGBFS, LE, LG	GBFS	distributed	Kuroiwa and Fukunaga 2019

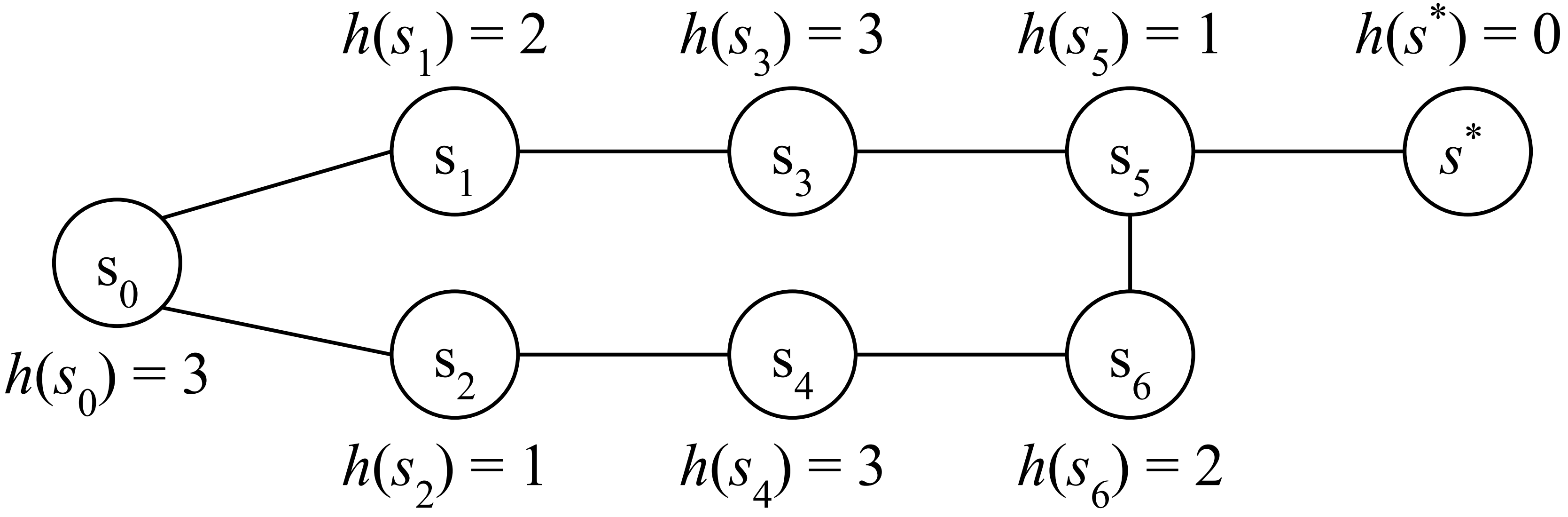
Parallel Best-First Search (BFS)

- BFS is used to solve graph search problems such as planning
- Many parallelization methods have been previously proposed
- Previous parallel A* methods have been experimentally shown to scale well
- **What about parallel Greedy Best-First Search (GBFS)?**

Method	BFS	Environment	Citation
HDA*	A*	distributed	Kishimoto et al. 2009
PBNF	A*, WA*	multi-core	Burns et al. 2010
KPBFS	WA*	multi-core	Vidal et al. 2010
HDGBFS, LE, LG	GBFS	distributed	Kuroiwa and Fukunaga 2019

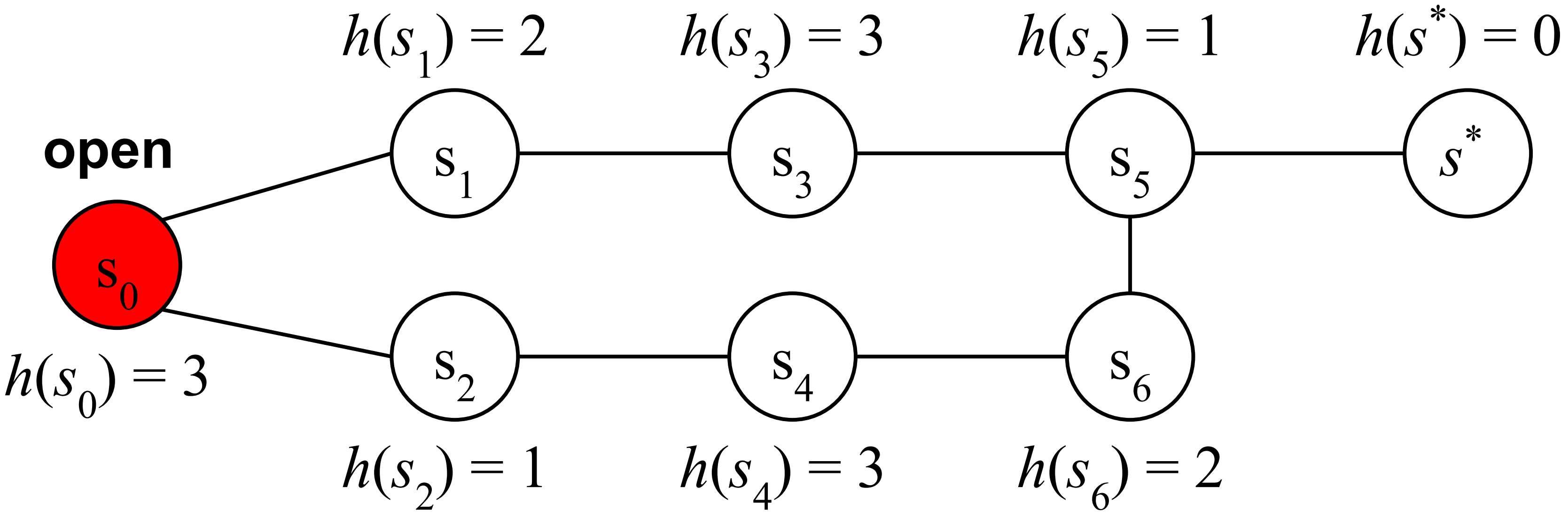
A Graph Search Problem Example with Greedy Best-First Search (GBFS)

- A solution of a graph search problem is a path from s_0 to s^*
- GBFS expands s with min. $h(s)$ (a heuristic value)



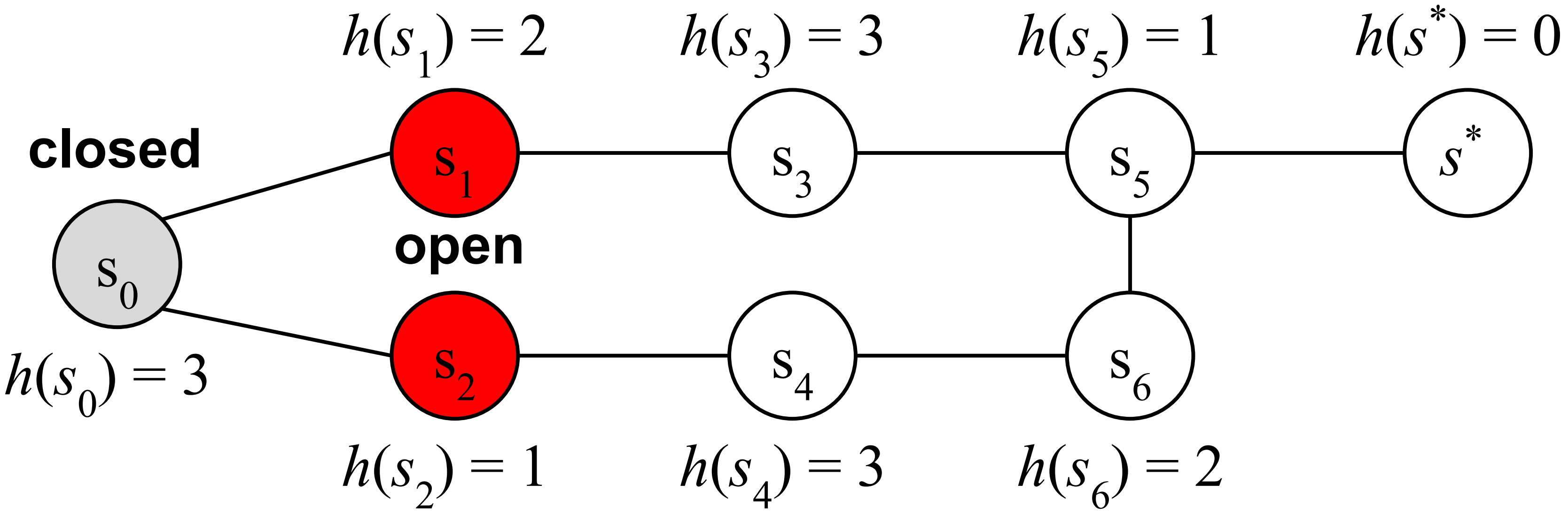
A Graph Search Problem Example with Greedy Best-First Search (GBFS)

- A solution of a graph search problem is a path from s_0 to s^*
- GBFS expands s with min. $h(s)$ (a heuristic value)



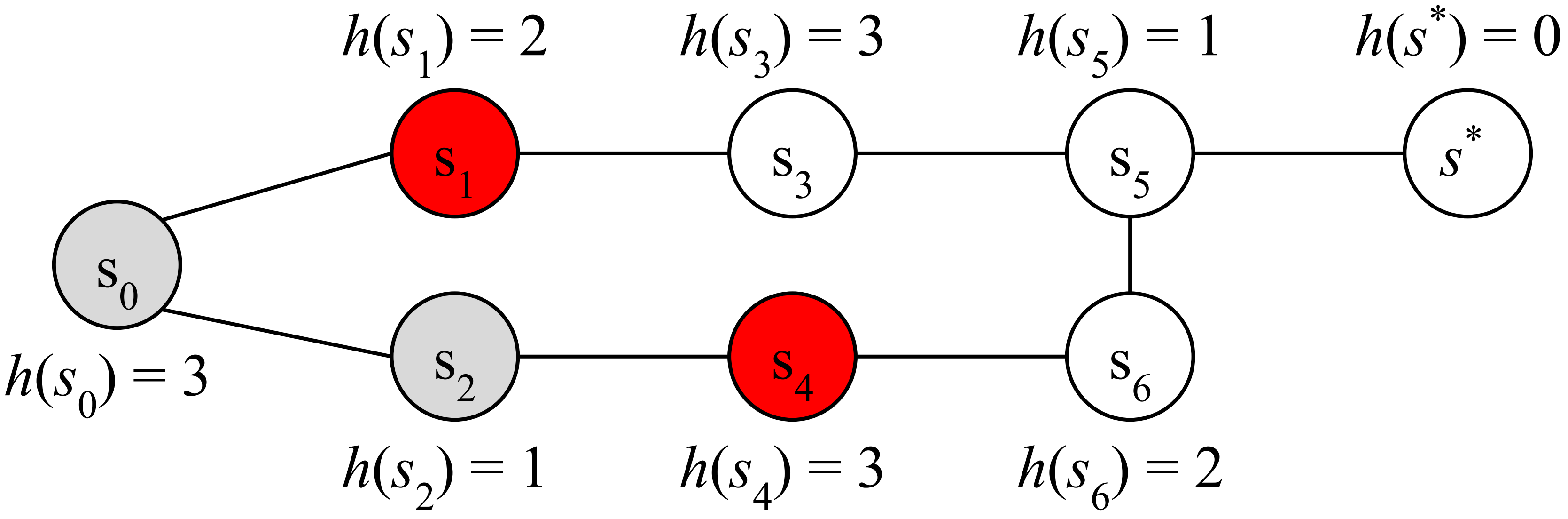
A Graph Search Problem Example with Greedy Best-First Search (GBFS)

- A solution of a graph search problem is a path from s_0 to s^*
- GBFS expands s with min. $h(s)$ (a heuristic value)



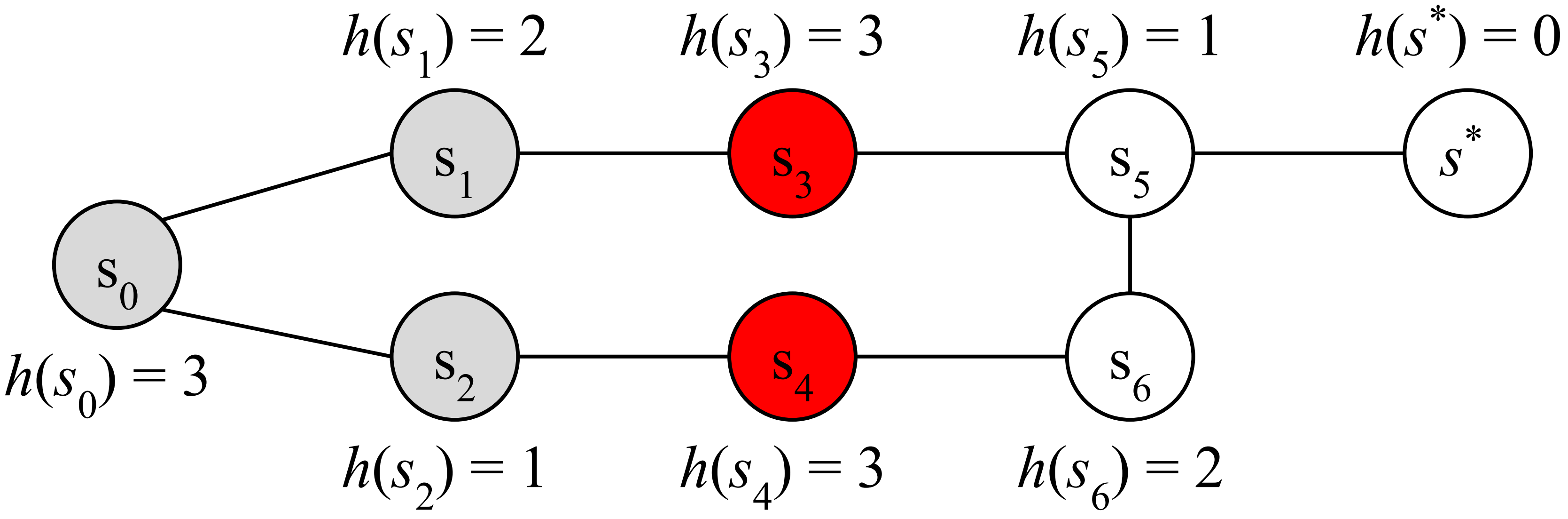
A Graph Search Problem Example with Greedy Best-First Search (GBFS)

- A solution of a graph search problem is a path from s_0 to s^*
- GBFS expands s with min. $h(s)$ (a heuristic value)



A Graph Search Problem Example with Greedy Best-First Search (GBFS)

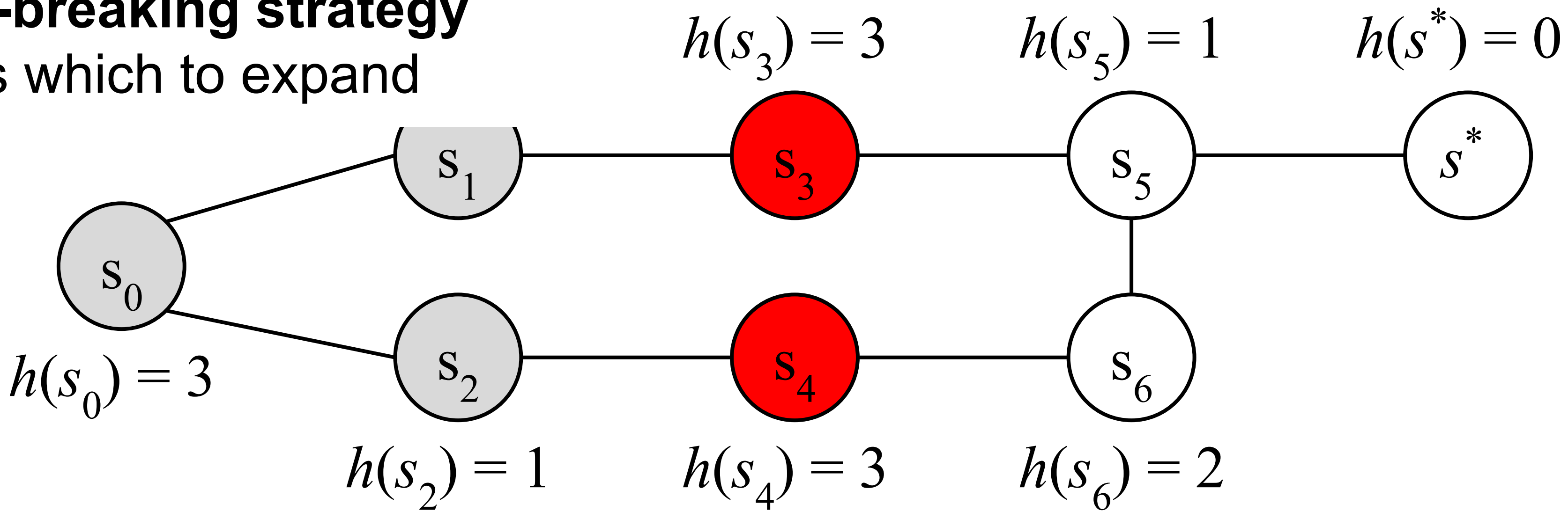
- A solution of a graph search problem is a path from s_0 to s^*
- GBFS expands s with min. $h(s)$ (a heuristic value)



A Graph Search Problem Example with Greedy Best-First Search (GBFS)

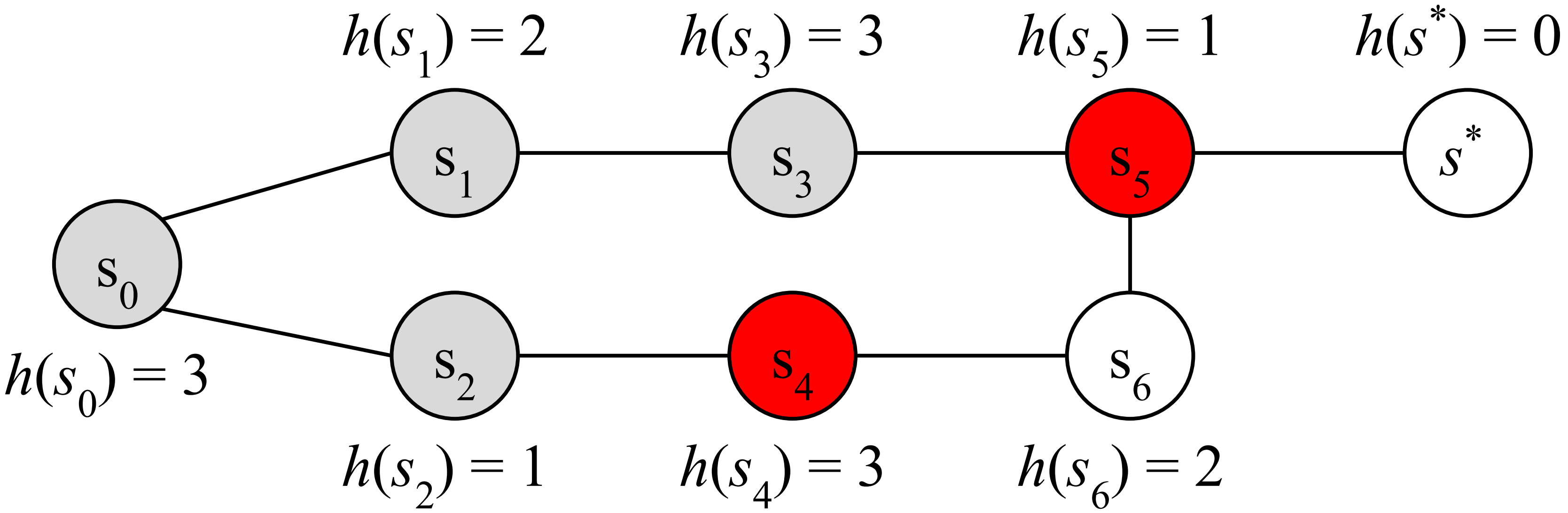
- A solution of a graph search problem is a path from s_0 to s^*
- GBFS expands s with min. $h(s)$ (a heuristic value)

The **tie-breaking strategy** decides which to expand



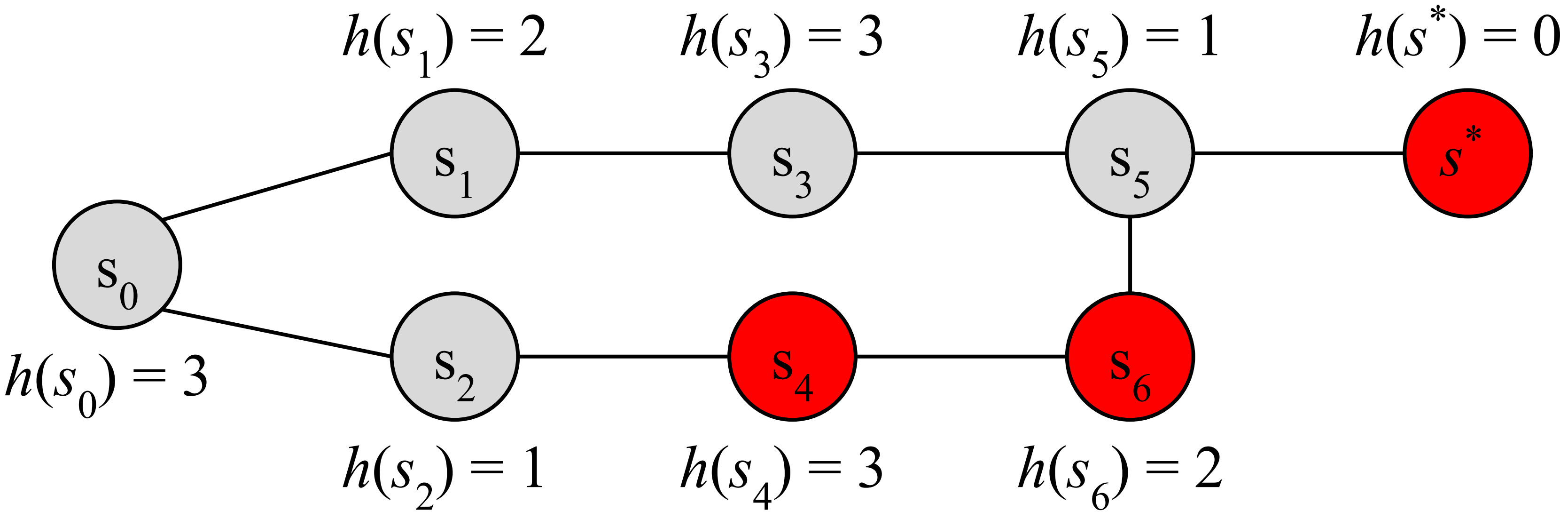
A Graph Search Problem Example with Greedy Best-First Search (GBFS)

- A solution of a graph search problem is a path from s_0 to s^*
- GBFS expands s with min. $h(s)$ (a heuristic value)



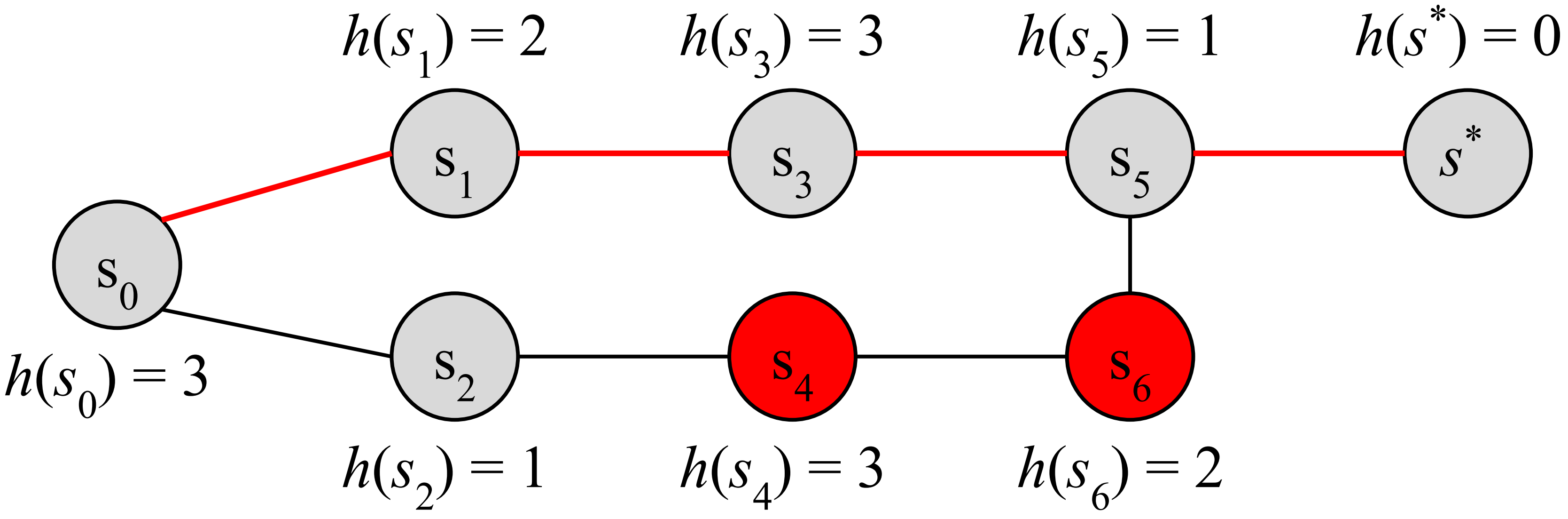
A Graph Search Problem Example with Greedy Best-First Search (GBFS)

- A solution of a graph search problem is a path from s_0 to s^*
- GBFS expands s with min. $h(s)$ (a heuristic value)



A Graph Search Problem Example with Greedy Best-First Search (GBFS)

- A solution of a graph search problem is a path from s_0 to s^*
- GBFS expands s with min. $h(s)$ (a heuristic value)



Pathological Behavior in Parallel GBFS
[Kuroiwa and Fukunaga 2019]

- GBFS, HDGBFS, LE, and LG were experimentally compared with the 5 min. time limit for each instance

Pathological Behavior in Parallel GBFS

[Kuroiwa and Fukunaga 2019]

- GBFS, HDGBFS, LE, and LG were experimentally compared with the 5 min. time limit for each instance
- Parallel GBFS methods **failed to solve easy instances** which GBFS solved within 1 sec. and 100 expansions

Pathological Behavior in Parallel GBFS
[Kuroiwa and Fukunaga 2019]

- GBFS, HDGBFS, LE, and LG were experimentally compared with the 5 min. time limit for each instance
- Parallel GBFS methods **failed to solve easy instances** which GBFS solved within 1 sec. and 100 expansions
- Parallel GBFS methods **expanded >1000 times as many states** as GBFS in some instances

Pathological Behavior in Parallel GBFS

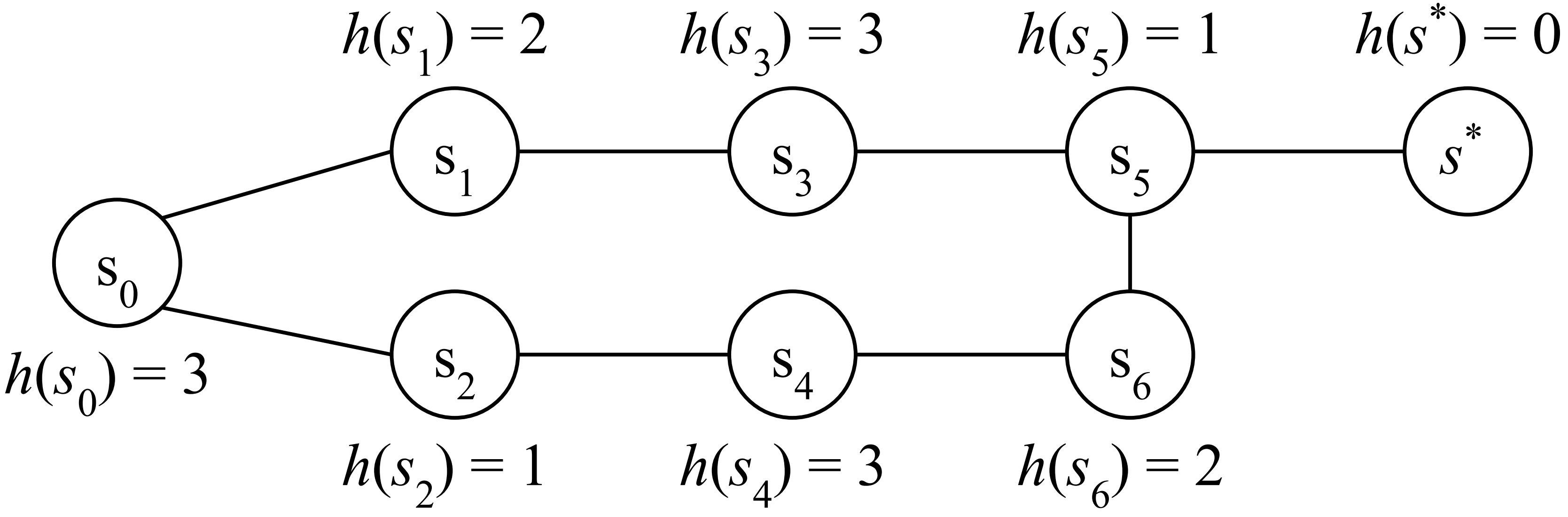
[Kuroiwa and Fukunaga 2019]

- GBFS, HDGBFS, LE, and LG were experimentally compared with the 5 min. time limit for each instance
- Parallel GBFS methods **failed to solve easy instances** which GBFS solved within 1 sec. and 100 expansions
- Parallel GBFS methods **expanded >1000 times as many states** as GBFS in some instances
- **Can we obtain theoretical bound for the performance degradation?**

KBFS: a Model of Parallel BFS

[Kuroiwa and Fukunaga 2019]

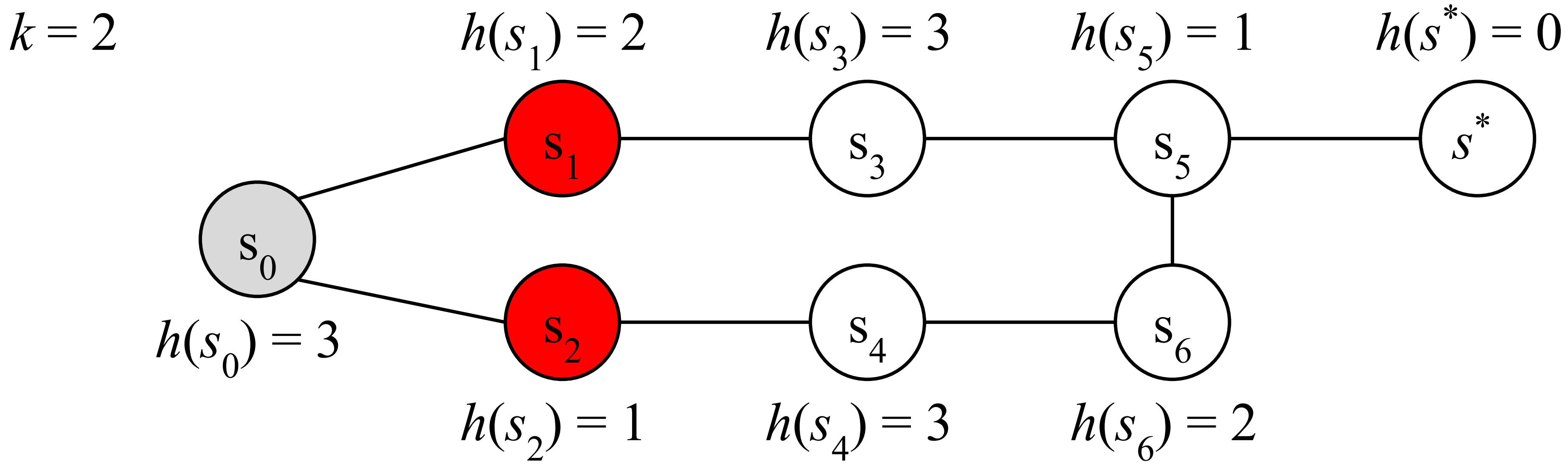
- KBFS [Felner et al. 2003]: similar to BFS, but simultaneously expands k states
- HDA*, KPBFS, HDGBFS, and LE can be modeled as KBFS



KBFS: a Model of Parallel BFS

[Kuroiwa and Fukunaga 2019]

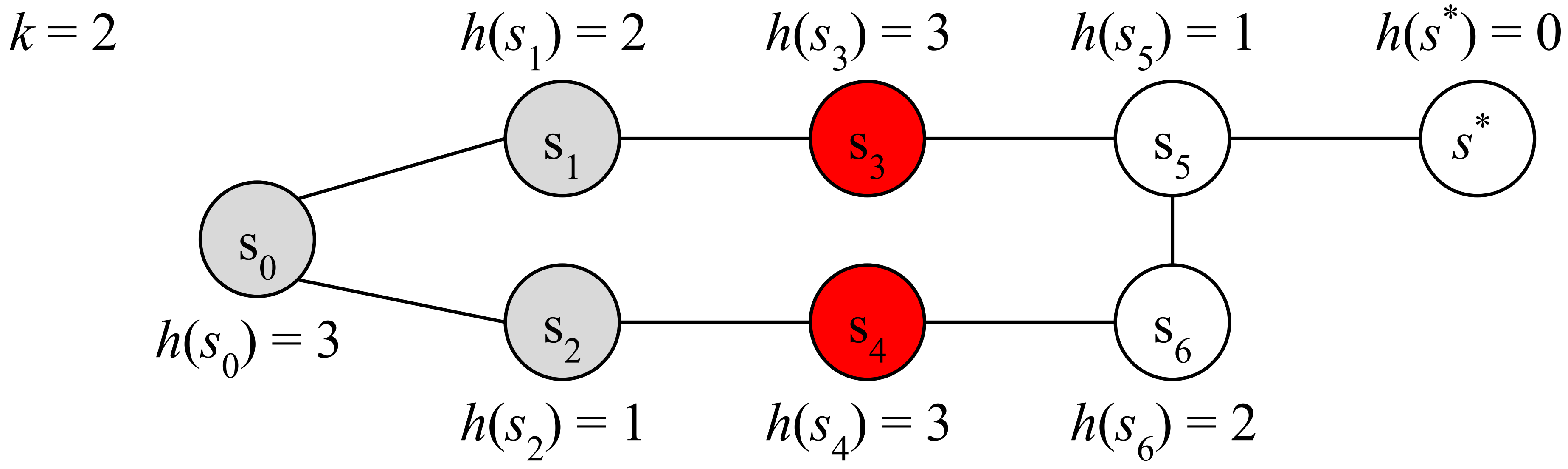
- KBFS [Felner et al. 2003]: similar to BFS, but simultaneously expands k states
- HDA*, KPBFS, HDGBFS, and LE can be modeled as KBFS



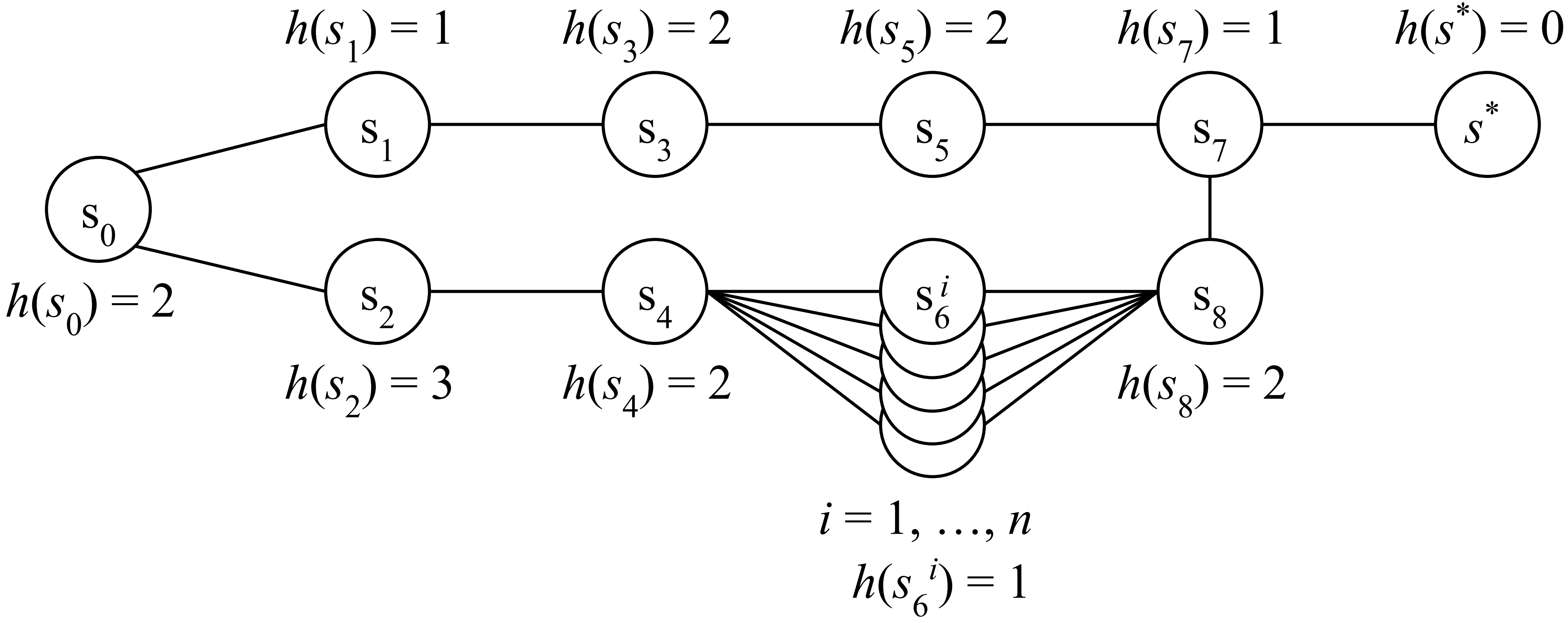
KBFS: a Model of Parallel BFS

[Kuroiwa and Fukunaga 2019]

- KBFS [Felner et al. 2003]: similar to BFS, but simultaneously expands k states
- HDA*, KPBFS, HDGBFS, and LE can be modeled as KBFS

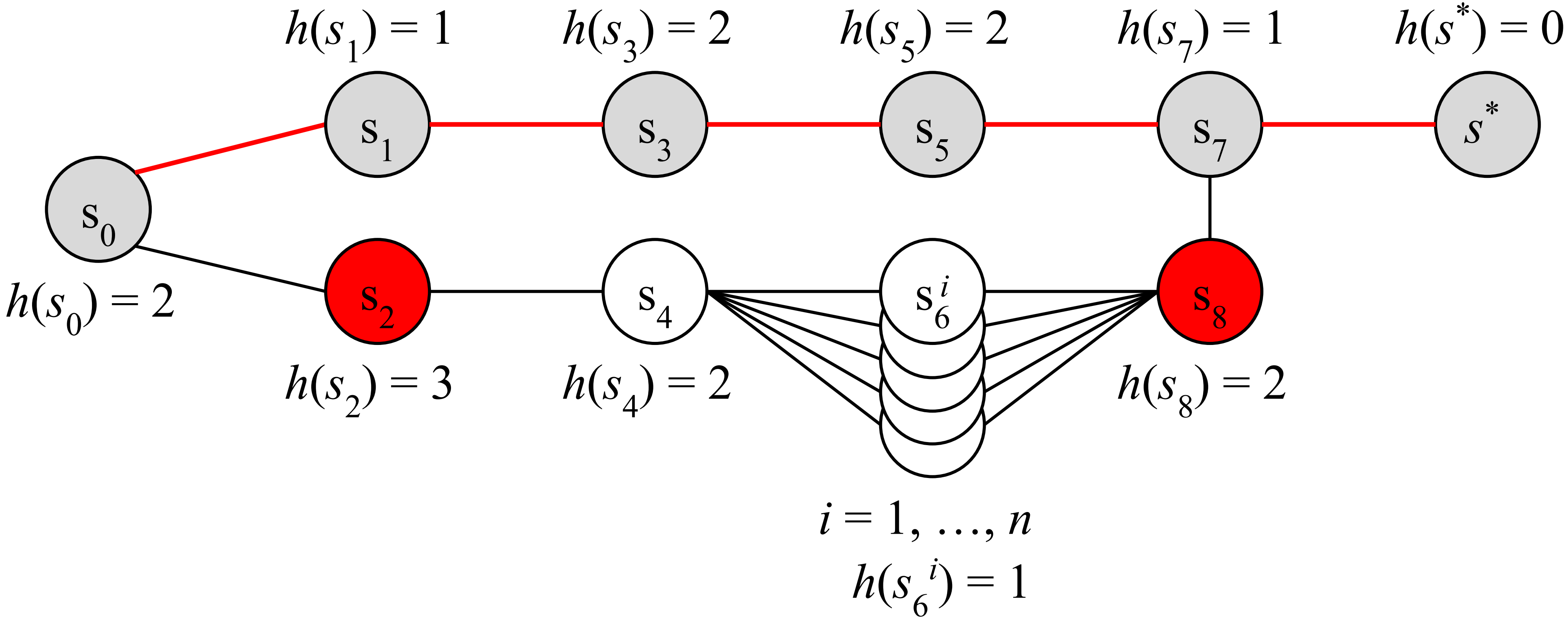


KGBFS (a GBFS Version of KBFS) can expand arbitrarily more states than GBFS



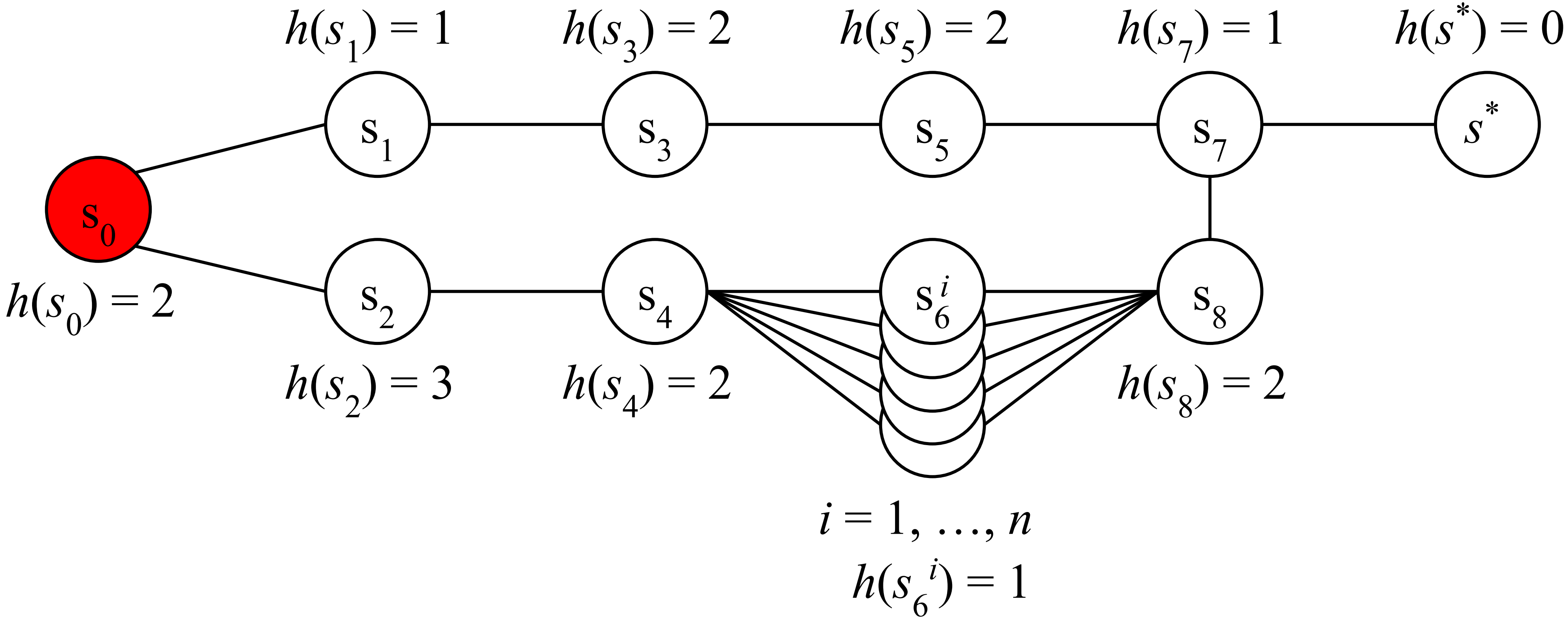
KGBFS (a GBFS Version of KBFS) can expand arbitrarily more states than GBFS

GBFS expands 6 **states**



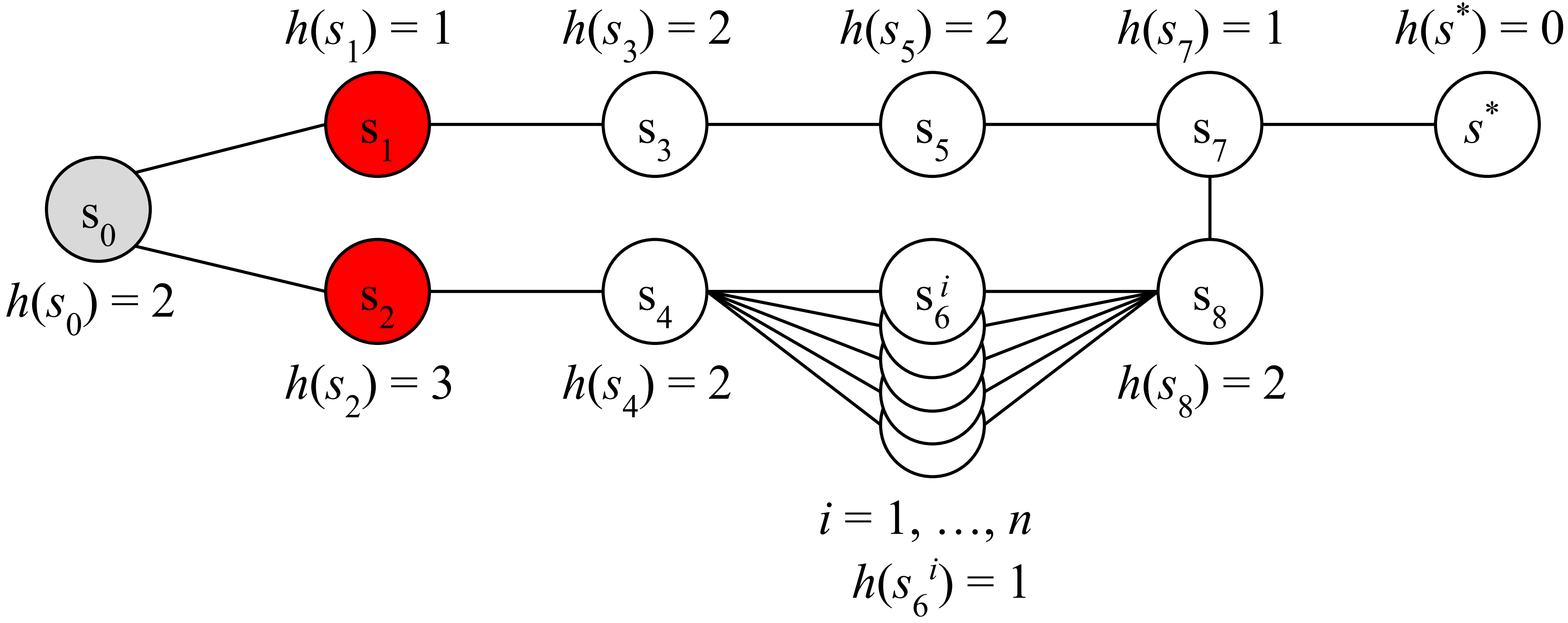
KGBFS (a GBFS Version of KBFS) can expand arbitrarily more states than GBFS

KGBFS with $1 < k \leqq n$



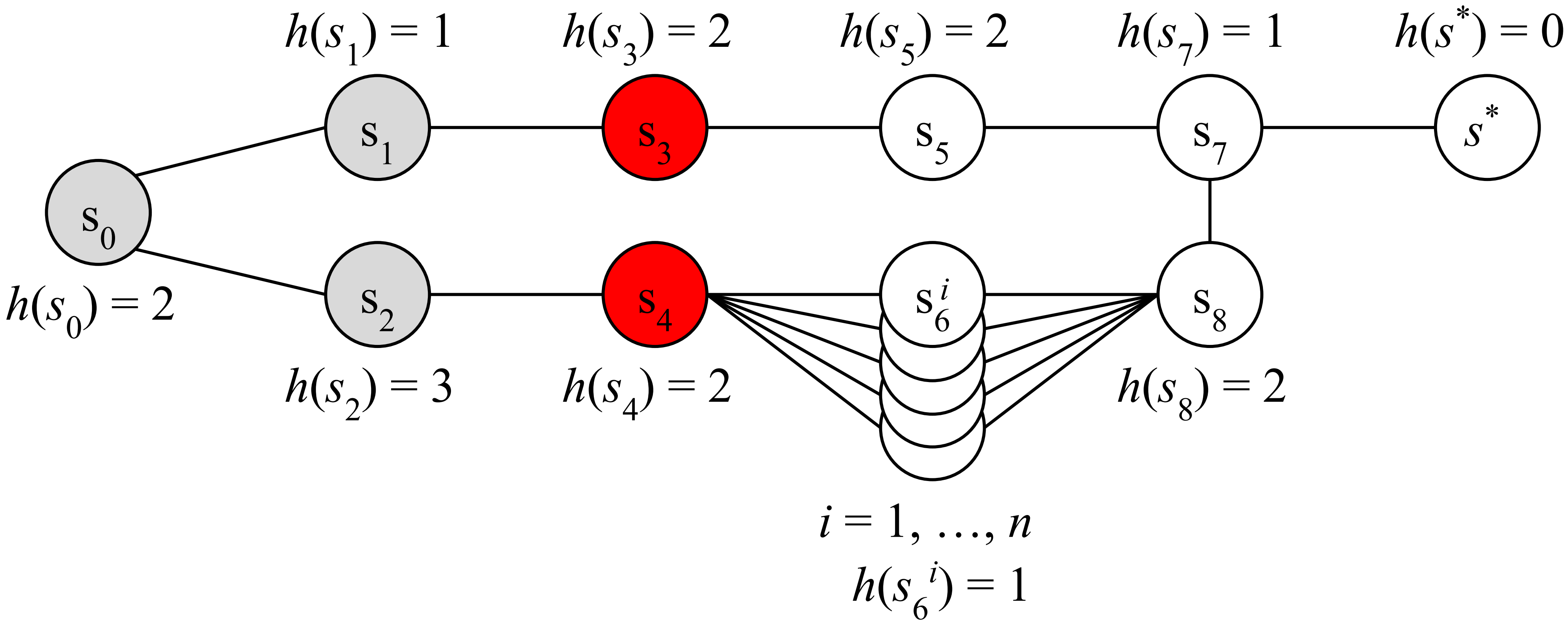
KGBFS (a GBFS Version of KBFS) can expand arbitrarily more states than GBFS

KGBFS with $1 < k \leqq n$



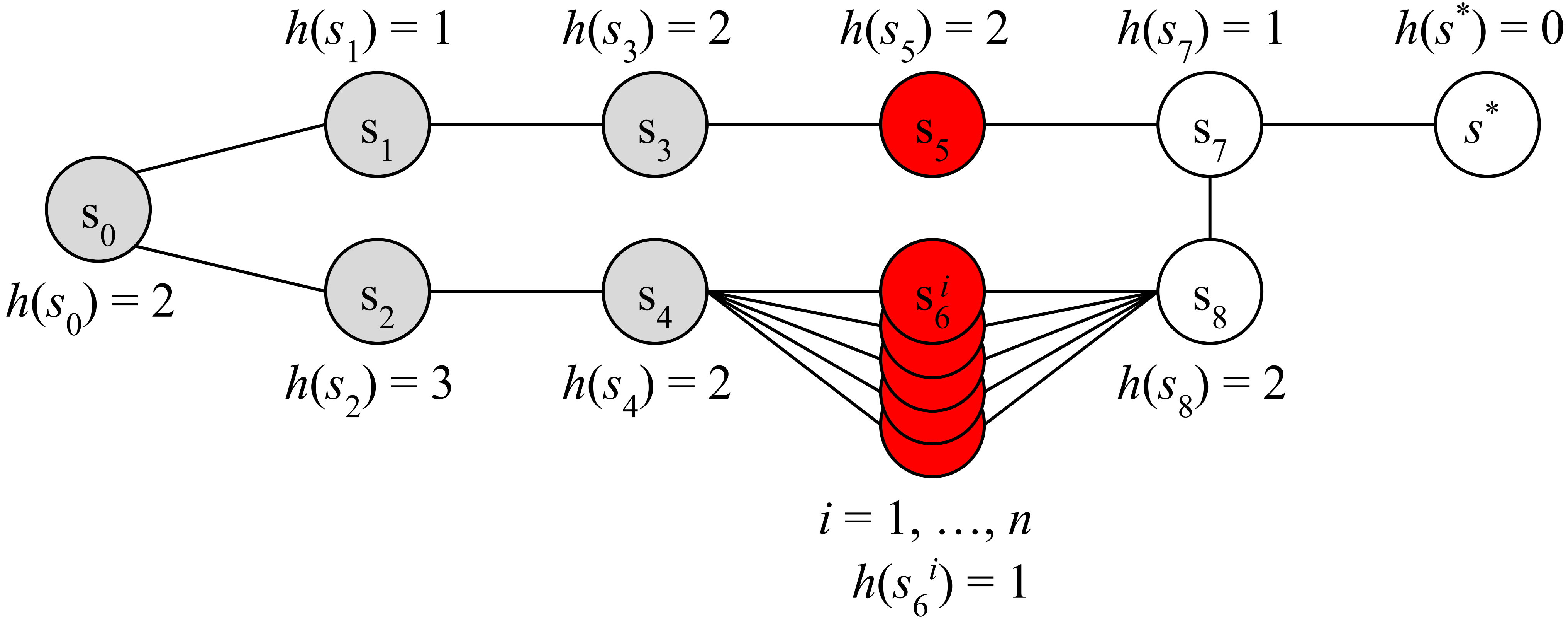
KGBFS (a GBFS Version of KBFS) can expand arbitrarily more states than GBFS

KGBFS with $1 < k \leqq n$



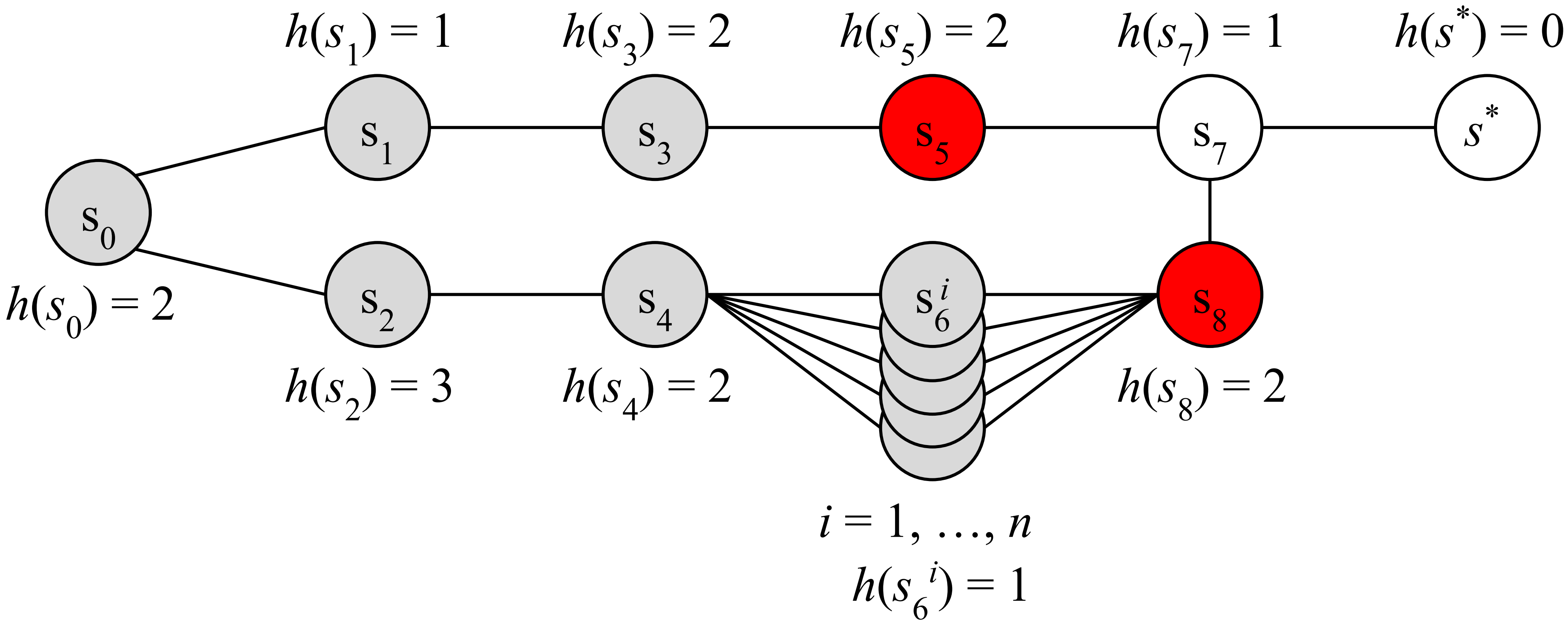
KGBFS (a GBFS Version of KBFS) can expand arbitrarily more states than GBFS

KGBFS with $1 < k \leqq n$



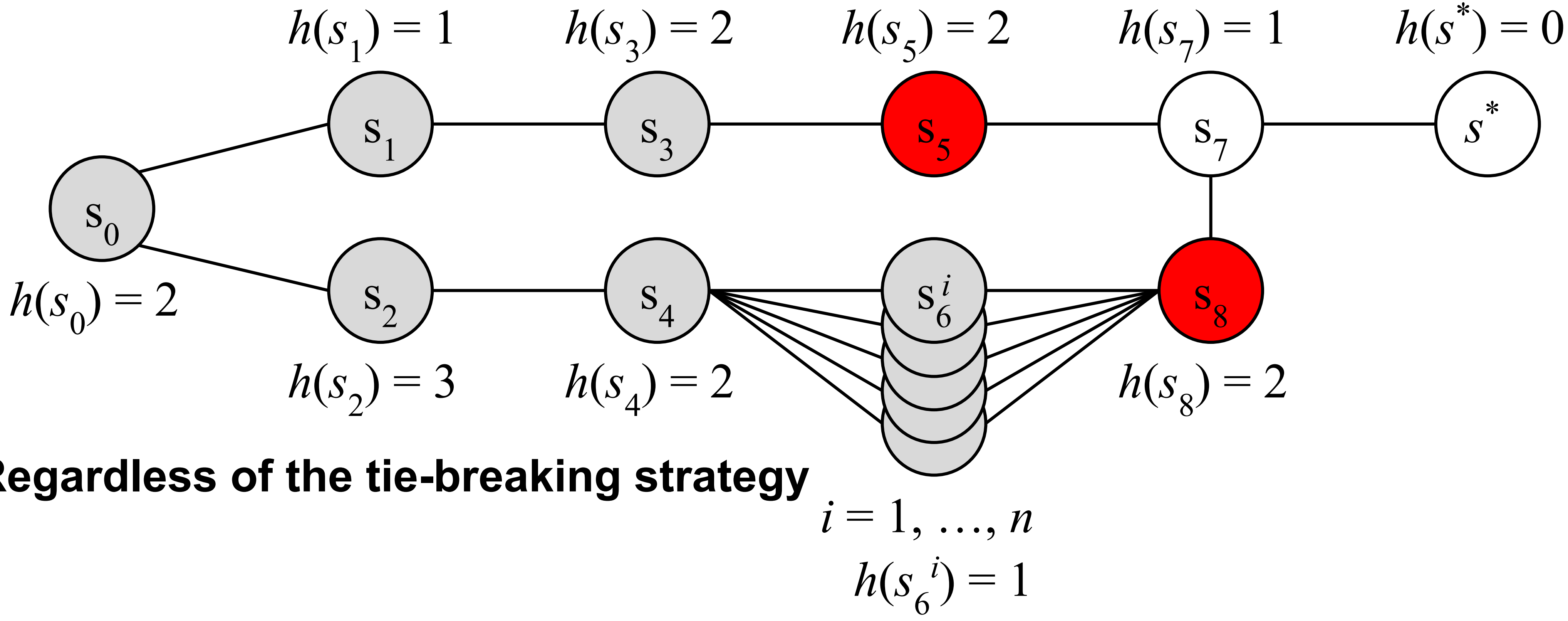
KGBFS (a GBFS Version of KBFS) can expand arbitrarily more states than GBFS

KGBFS with $1 < k \leqq \mathrm{n}$ **expands more than** n **states**: $n \to \infty$



KGBFS (a GBFS Version of KBFS) can expand arbitrarily more states than GBFS

KGBFS with $1 < k \leqq \mathrm{n}$ **expands more than** n **states**: $n \to \infty$



Regardless of the tie-breaking strategy

Pathology and t-Boundedness

- A is **pathological** relative to B if A expands arbitrarily more states than B does given some graph and heuristic
- A is ***t*-bounded** relative to B if A expands no more than t times as many states as B does, for any graph and heuristic

Pathologies in Parallel BFS

Method	**Heuristic**	***t*-boundedness**	**Model**	**BFS**
HDA* (k processes)	consistent	k-bounded	KBFS (KA*)	A*
HDA*	inconsistent	pathological	KBFS (KA*)	A*
KPBFS ($w > 1$)		pathological	KBFS (KWA*)	WA*
KPBFS, HDGBFS, LE		pathological	KGBFS	GBFS
LG		pathological	KGBFS-like	GBFS

KA* is k-Bounded Relative to A* with a consistent heuristic

- A heuristic is **consistent**: $h(s') \geqq h(s) + c(s, s')$
  $c(s, s')$ is the cost of the edge (s, s')
- KA* with any tie-breaking strategy is k-bounded relative to A* with **the worst-case tie-breaking strategy**

Proof Sketch: A* expands s with min. $f(s) = g(s) + h(s)$ where $g(s)$ is the cost of the path from s_0 to s. A* expands each s with $f(s) < f^*$ (the cost of the optimal path). KA* expands s with $f(s) \leqq f^*$ at every k expansions because s' on an optimal path has $f(s') \leqq f^*$ by the consistency.

TB-Boundedness: Another Type of Bound

A is ***TB*-bounded** relative to B if A expands only states expanded by B with some tie-breaking strategy

Method	***TB*-boundedness**	**BFS**
HDA*	no	A*
KPBFS ($w > 1$)	no	WA*
KPBFS, HDGBFS, LE, LG	no	GBFS

P_{GBFS}/C: a k-Bounded Parallel GBFS

- P_{GBFS} executes k threads of independent GBFS with different tie-breaking strategies in parallel [Kuroiwa and Fukunaga 2019]
  Each GBFS does not change its expansion order
- P_{GBFS}/C uses the **shared evaluation cache** of heuristic values to speed up each GBFS while keeping the expansion order
- P_{GBFS} and P_{GBFS}/C are also TB-bounded

SPUHF: a TB-Bounded Parallel GBFS

- PUHF (Parallel Under High-water mark First) expands s only if $h(s) \leqq h(parent(s))$ or any other thread is not expanding a state
- *Proof Sketch*: If $h(s) \leqq h(parent(s))$, s is expanded by GBFS with some tie-breaking strategy
- SPUHF (Speculative PUHF) executes independent parallel search (**speculative search**) using idling threads with the **shared evaluation cache**

Bounded Parallel GBFS

- P_{GBFS}/C: k-bounded and TB-bounded
  A parallel portfolio of independent GBFS with different tie-breaking strategies using the shared evaluation cache
  k-bounded and TB-bounded
- SPUHF: TB-bounded
  A multi-core parallel GBFS similar to KPBFS
- See the paper for details

Experimental Results

Method	Coverage	# of solved instances unsolved by GBFS	# of unsolved instances solved by GBFS
LG	888	137	13
KPBFS	880	135	19
P_{GBFS}/C	**928**	164	**0**
SPUHF	864	115	15

Domain-wise results are complementary

55 domains from IPC-98-18
5 min. time limit
122 GiB memory limit

Conclusion/Summary of Contributions

- Proposed t-boundedness, Pathology, TB-boundedness
- Analyzed existing parallel BFS
- Proposed P_{GBFS}/C and SPUHF

Method	t-boundedness	TB-boundedness
HDA* [Kishimoto et al. 2009] (k processes), consistent heuristic	k-bounded	no
HDA*, inconsistent heuristic	pathological	no
KPBFS ($w > 1$) [Vidal et al. 2010]	pathological	no
KPBFS, HDGBFS, LE, LG [Kuroiwa and Fukunaga 2019]	pathological	no
P_{GBFS}/C [New]	k-bounded	TB-bounded
SPUHF [New]	unknown	TB-bounded