Graph Theory

Part Two

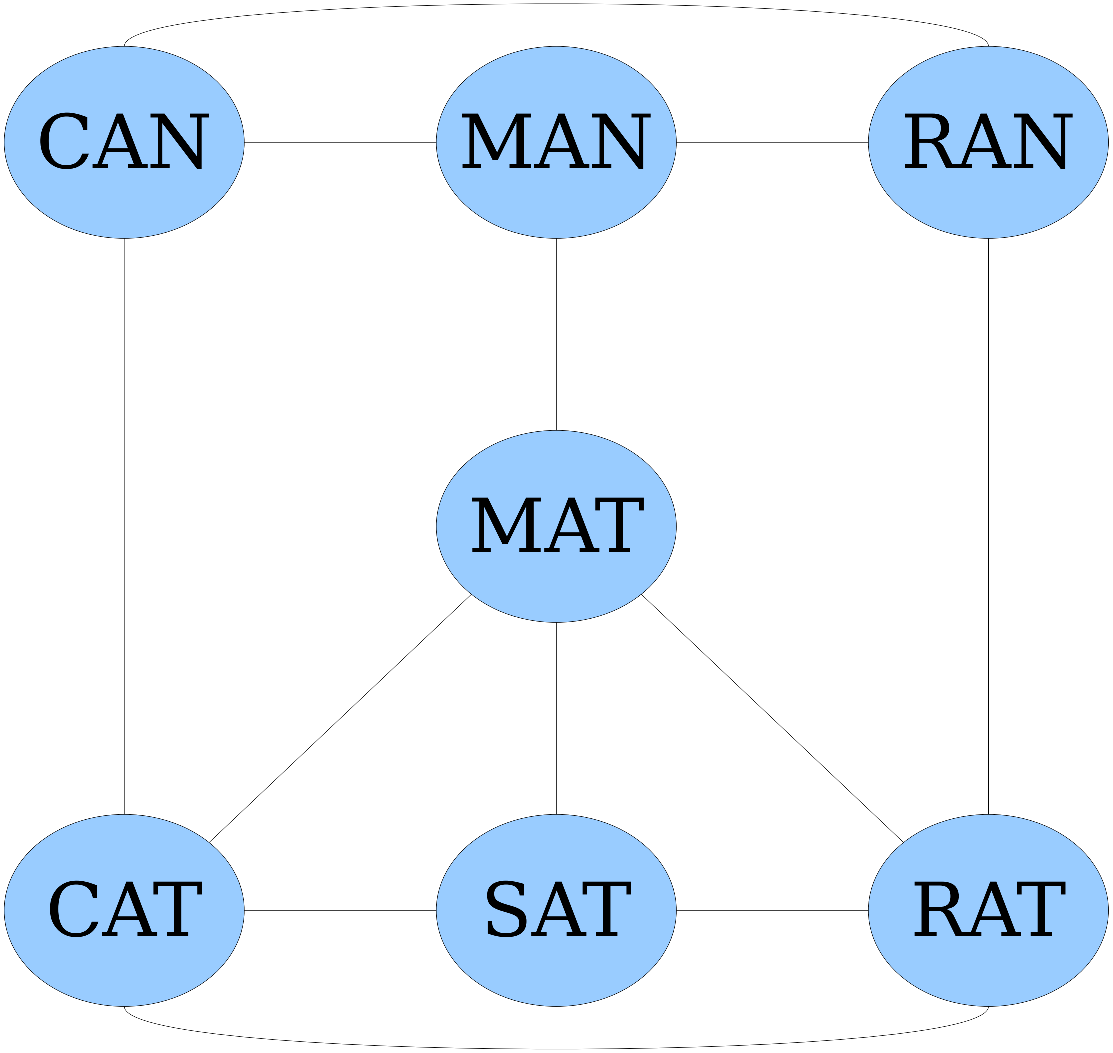
Outline for Today

- ***Walks, Paths, and Reachability***
  - Walking around a graph.
- ***The Teleported Train Problem***
  - A very exciting commute.
- ***Graph Complements***
  - Looking at negative space.

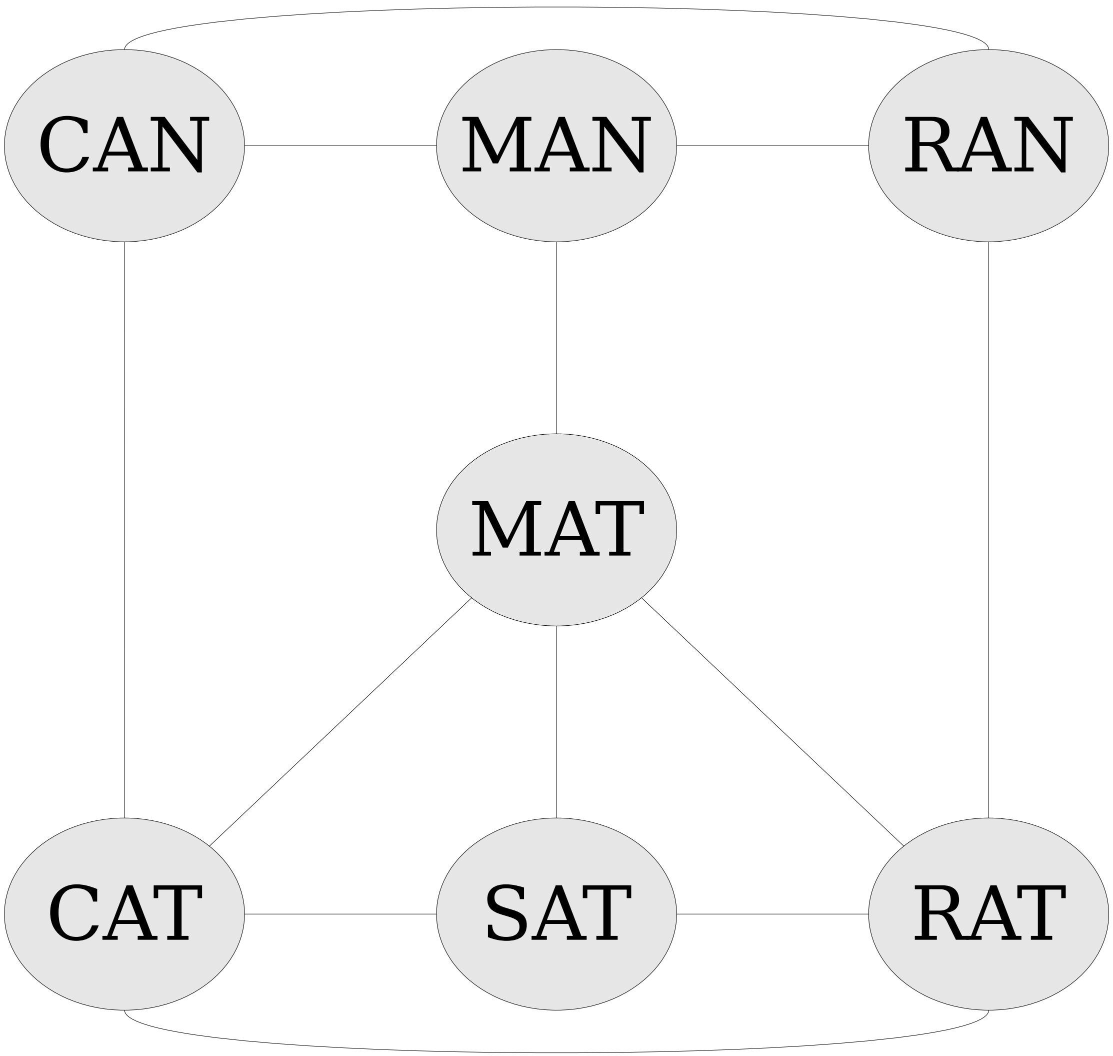
Recap from Last Time

Graphs and Digraphs

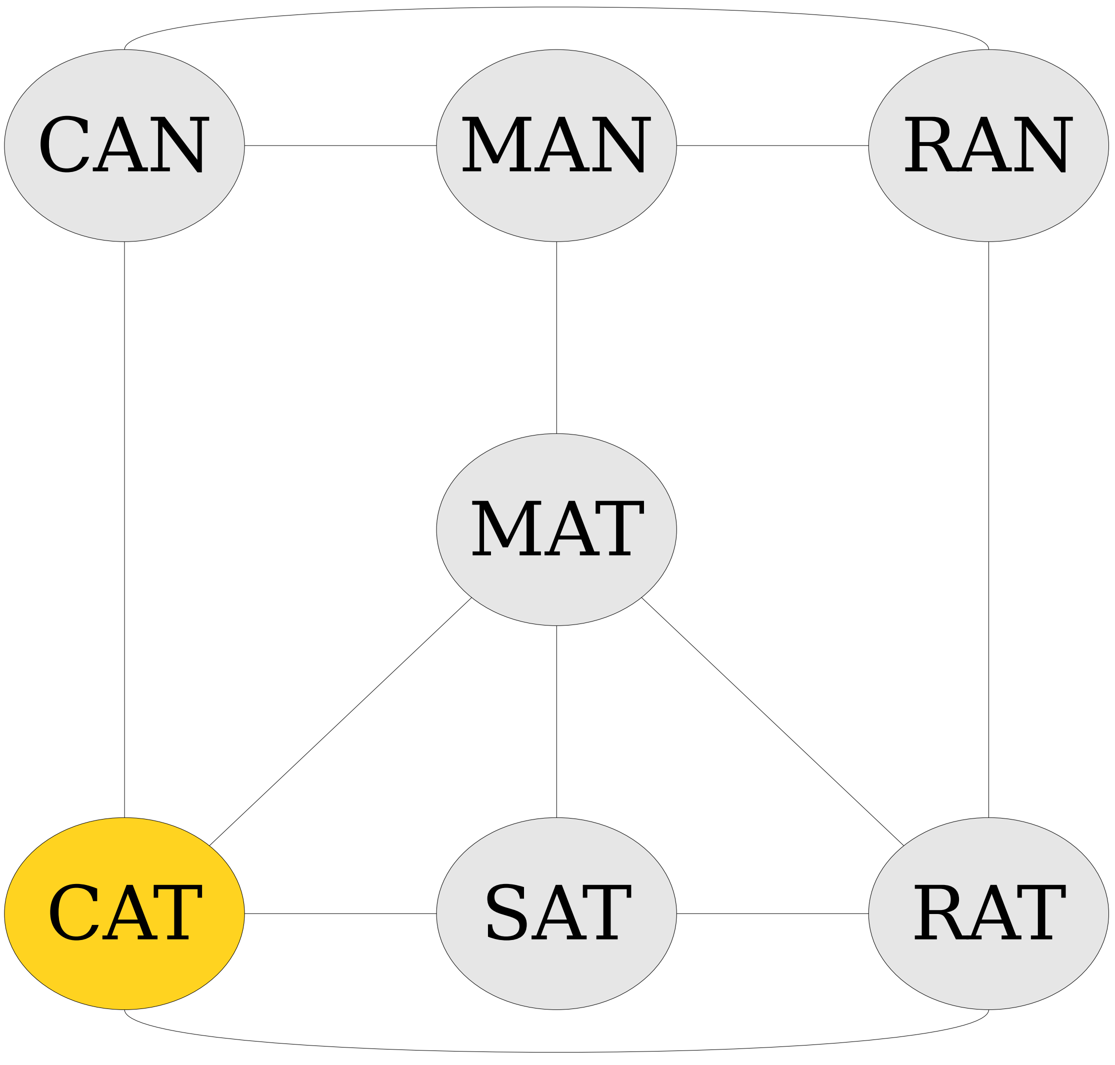
- A ***graph*** is a pair $G = (V, E)$ of a set of nodes V and set of edges E.
  - Nodes can be anything.
  - Edges are ***unordered pairs*** of nodes. If $\{u, v\} \in E$, then there's an edge from u to v.
- A ***digraph*** is a pair $G = (V, E)$ of a set of nodes V and set of directed edges E.
  - Each edge is represented as the ordered pair (u, v) indicating an edge from u to v.



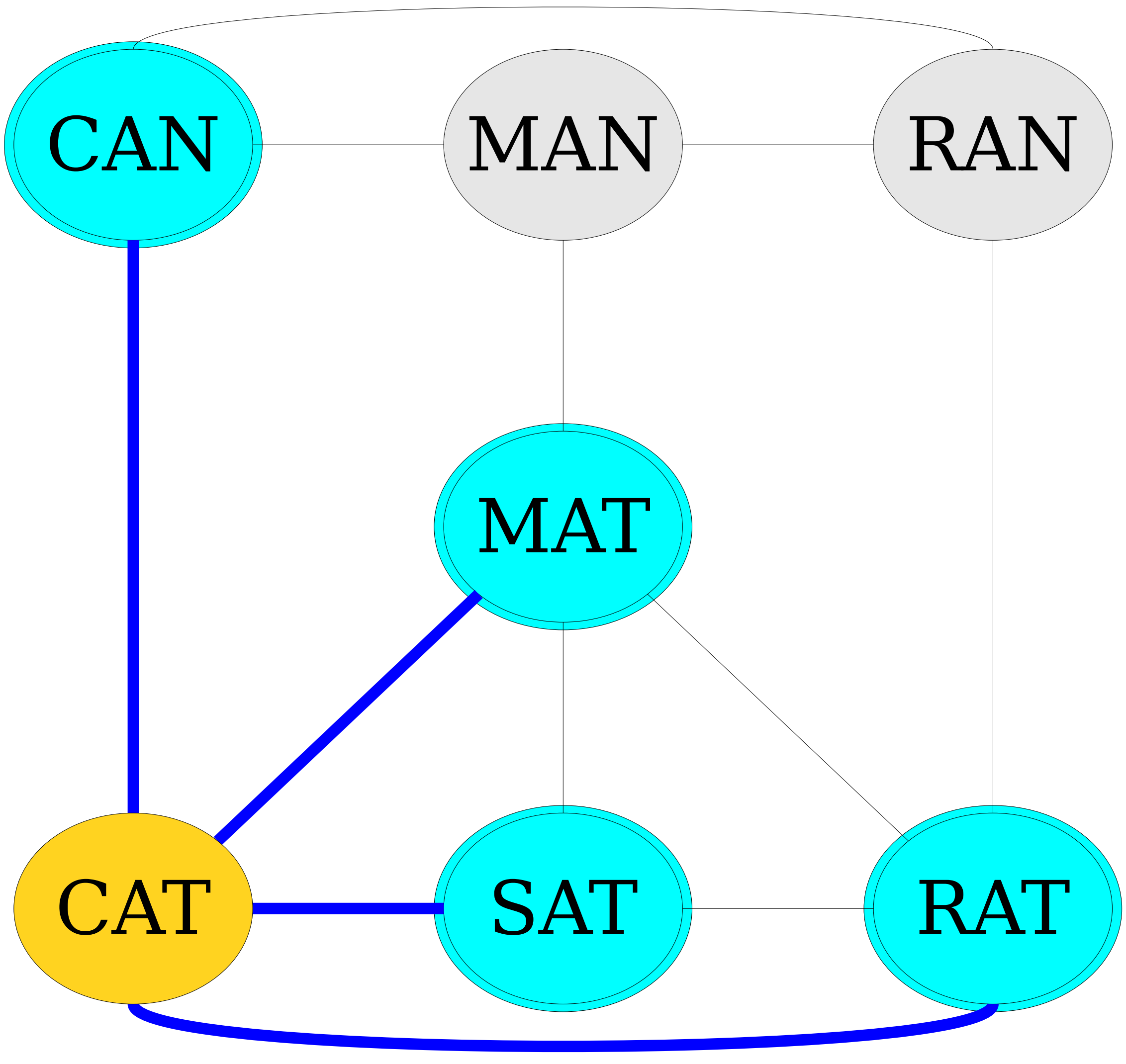
Two nodes in an undirected graph are called ***adjacent*** if there is an edge between them.



Two nodes in an undirected graph are called ***adjacent*** if there is an edge between them.



Two nodes in an undirected graph are called ***adjacent*** if there is an edge between them.



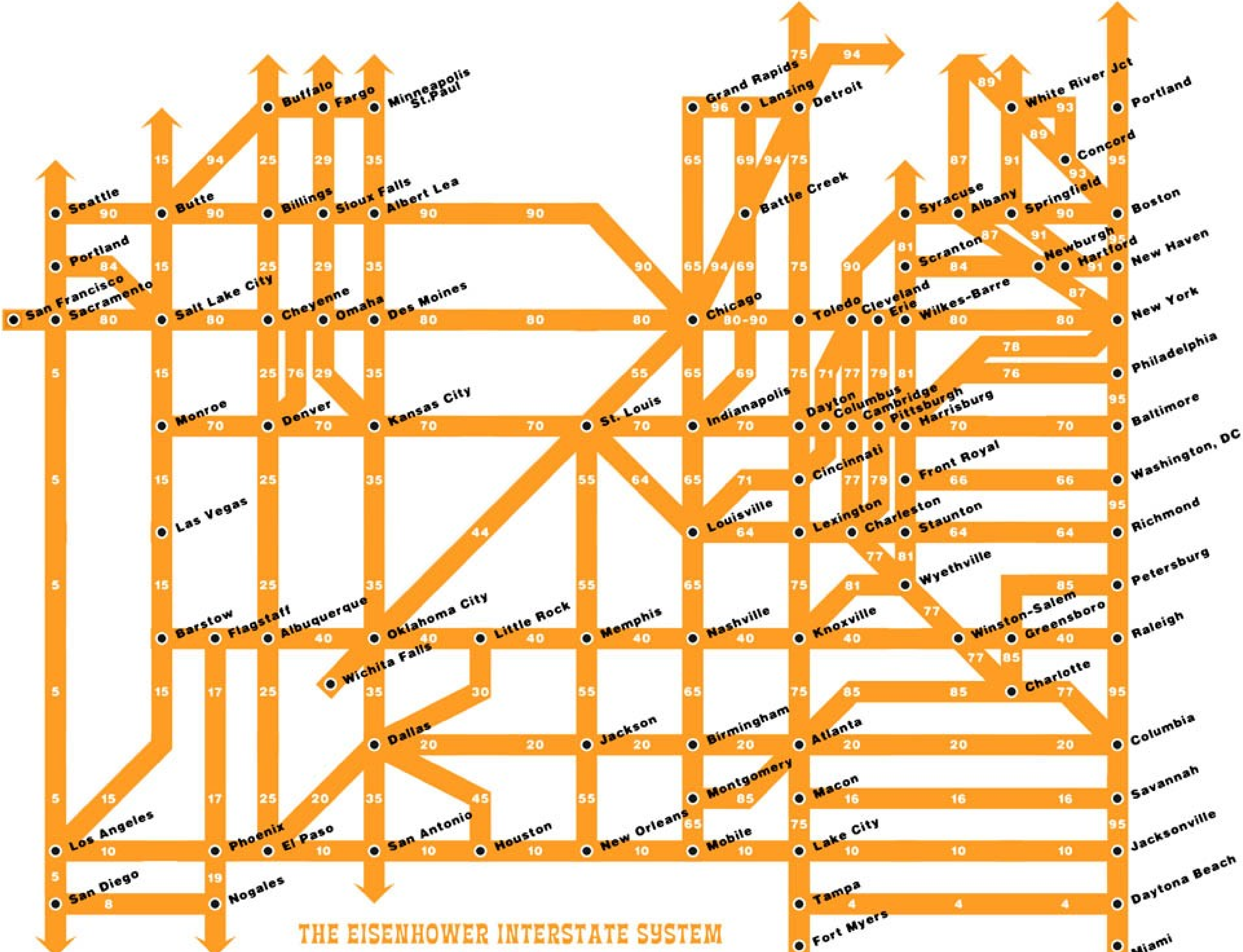
Two nodes in an undirected graph are called ***adjacent*** if there is an edge between them.

Using our Formalisms

- Let $G = (V, E)$ be an (undirected) graph.
- Intuitively, two nodes are adjacent if they're linked by an edge.
- Formally speaking, we say that two nodes $u, v \in V$ are ***adjacent*** if we have $\{u, v\} \in E$.
- There isn't an analogous notion for directed graphs. We usually just say "there's an edge from u to v" as a way of reading $(u, v) \in E$ aloud.

New Stuff!

Walks, Paths, and Reachability

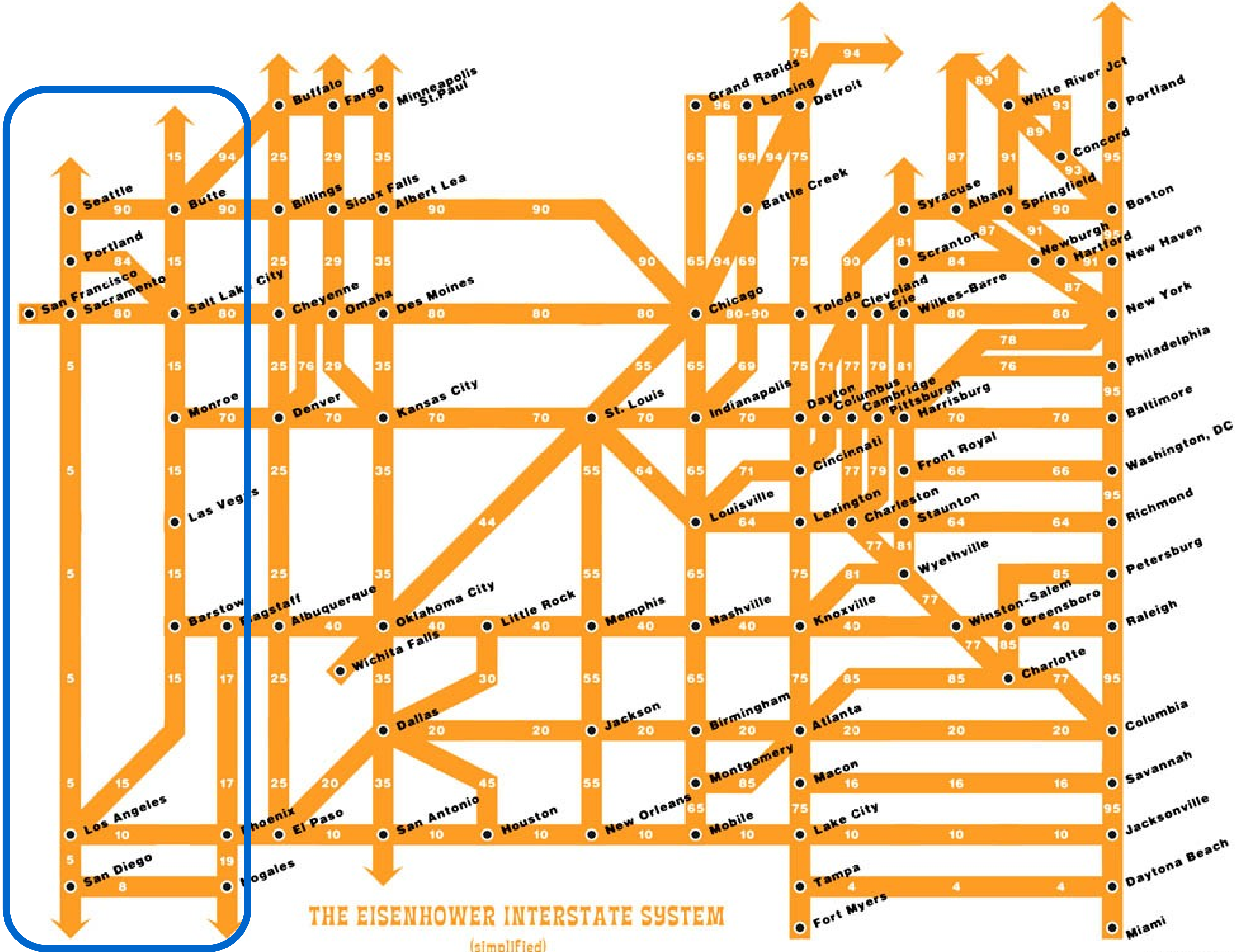


THE EISENHOWER INTERSTATE SYSTEM

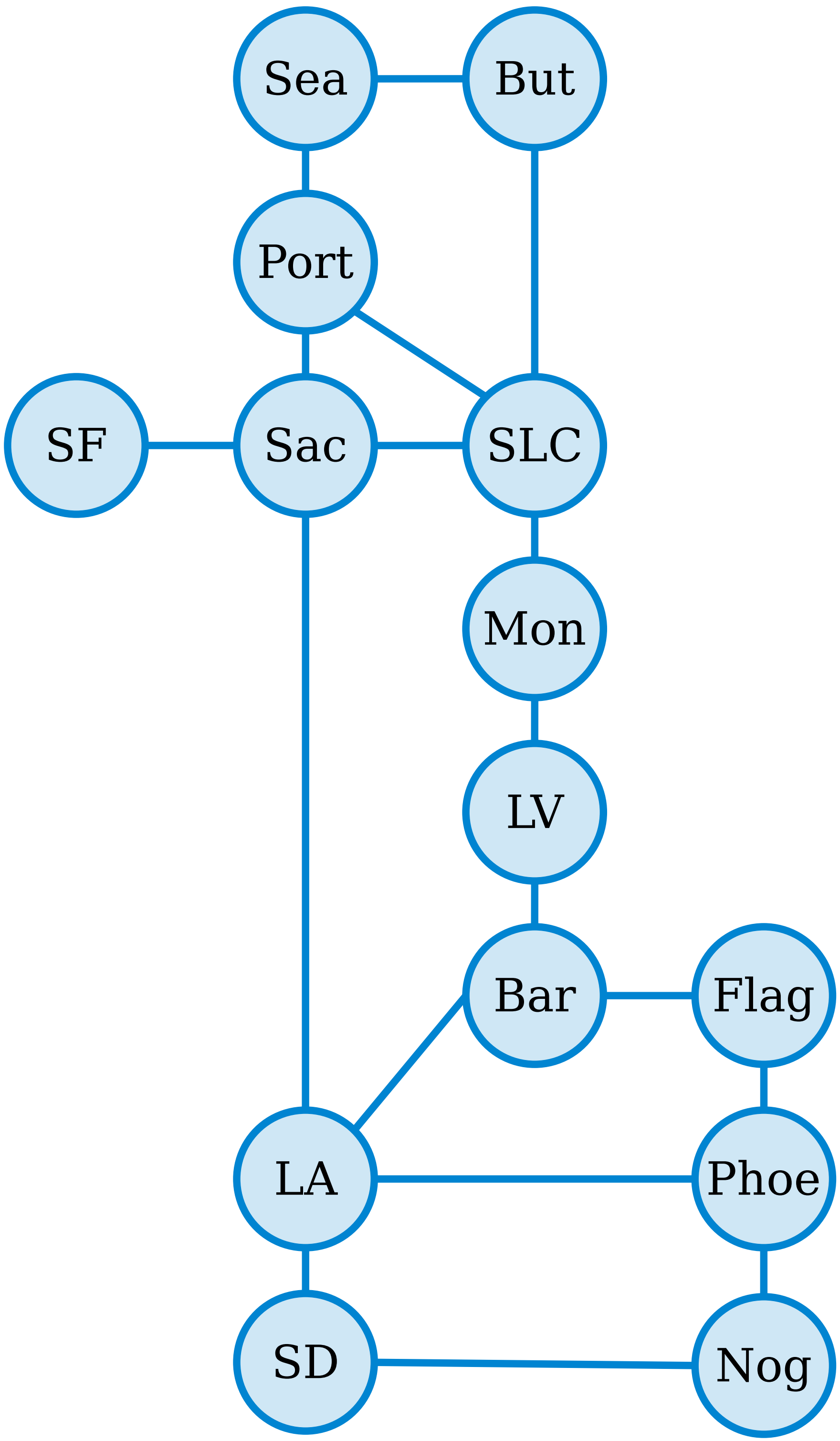
(simplified)

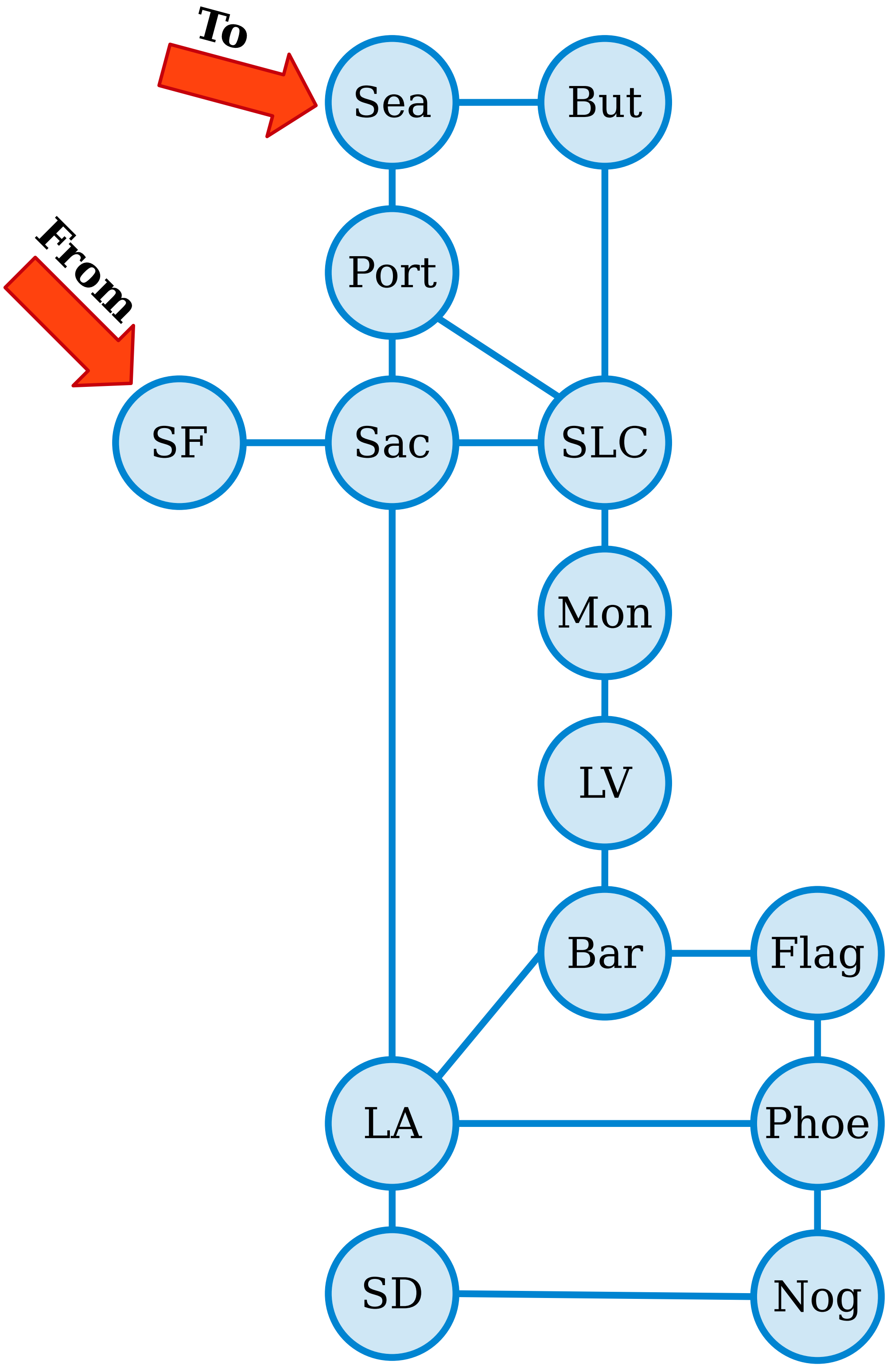
CHRIS YATES 2007

http://strangemaps.files.wordpress.com/2007/02/fullinterstatemap-web.jpg

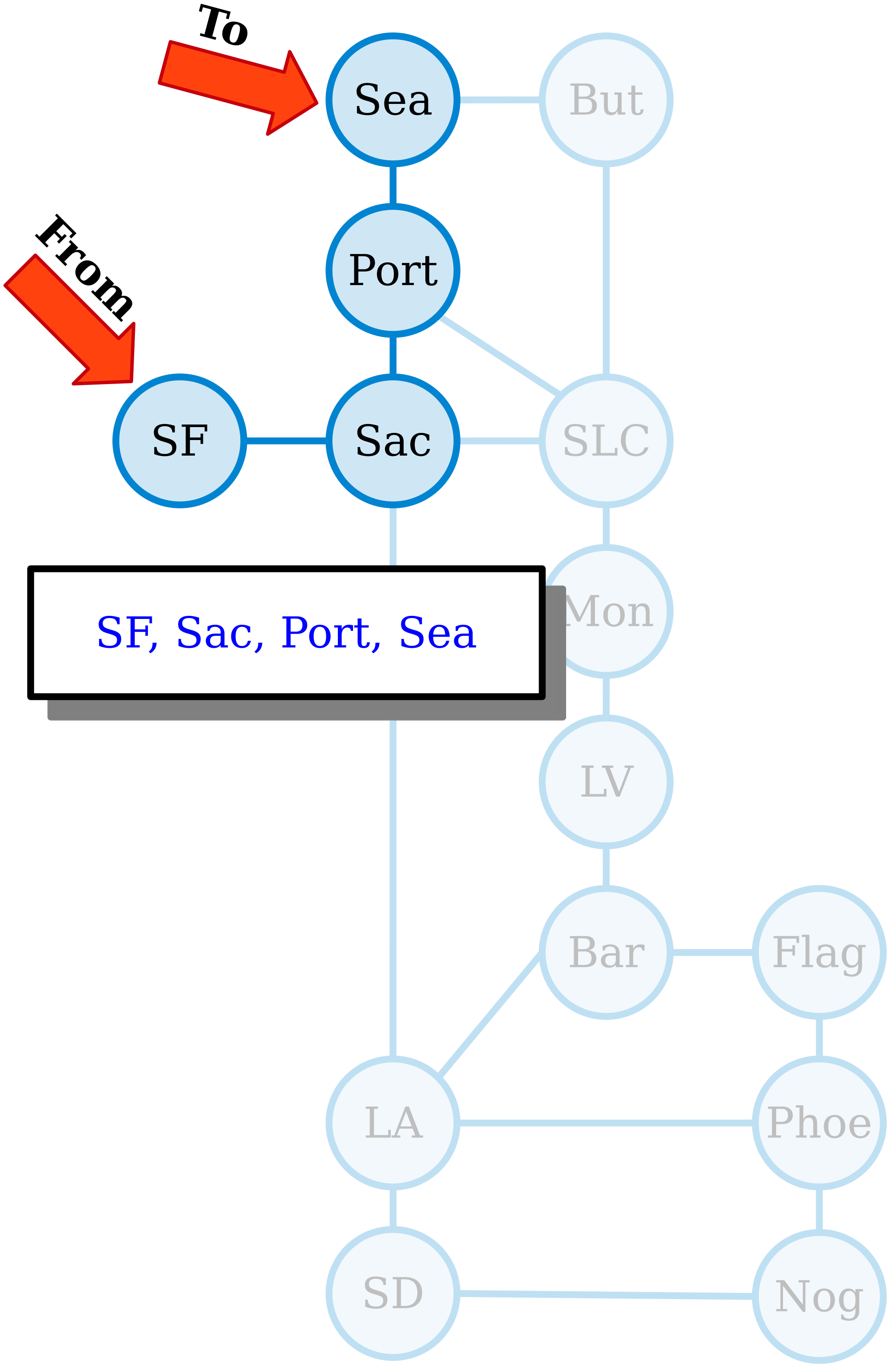


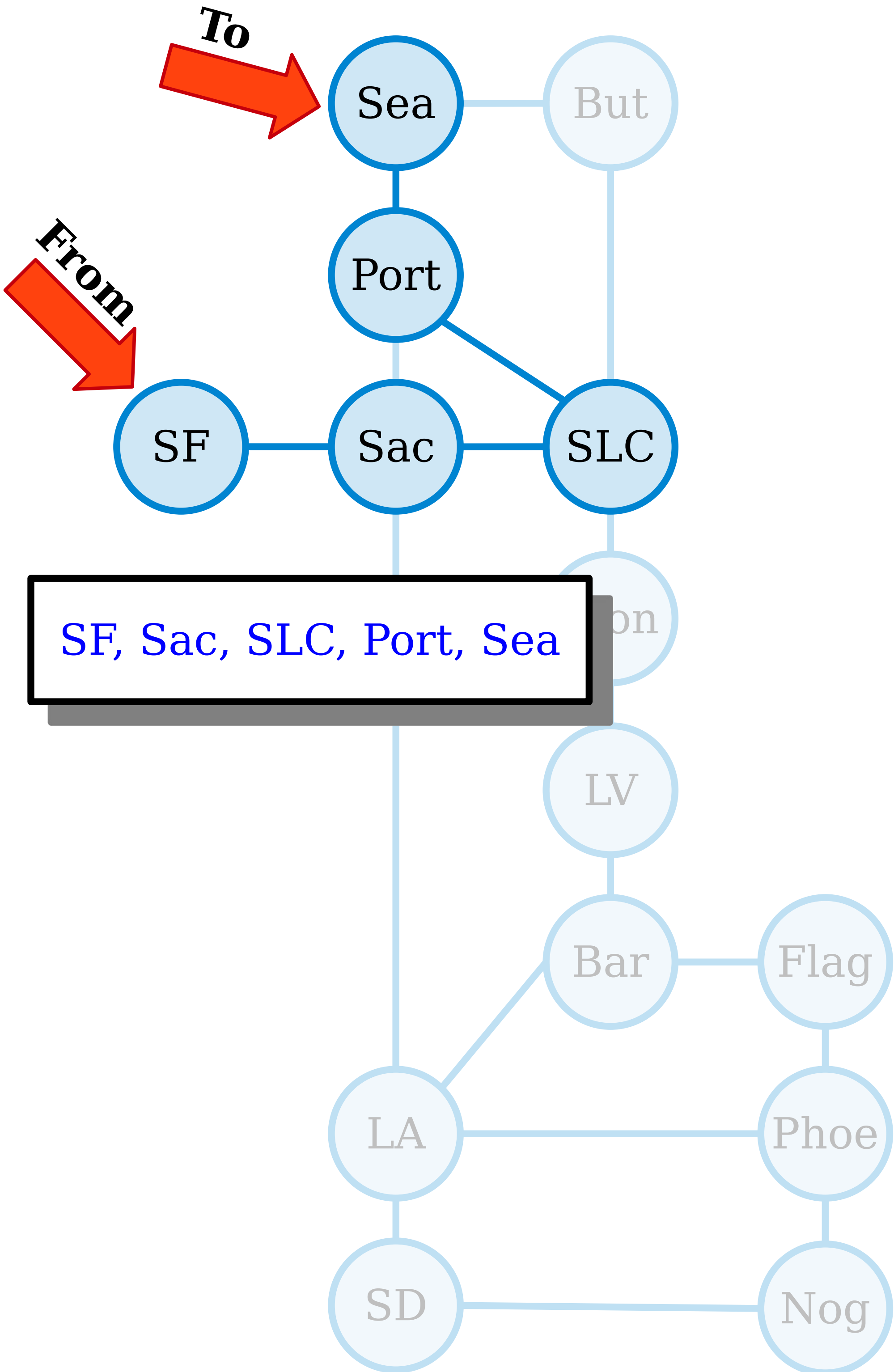
http://strangemaps.files.wordpress.com/2007/02/fullinterstatemap-web.jpg

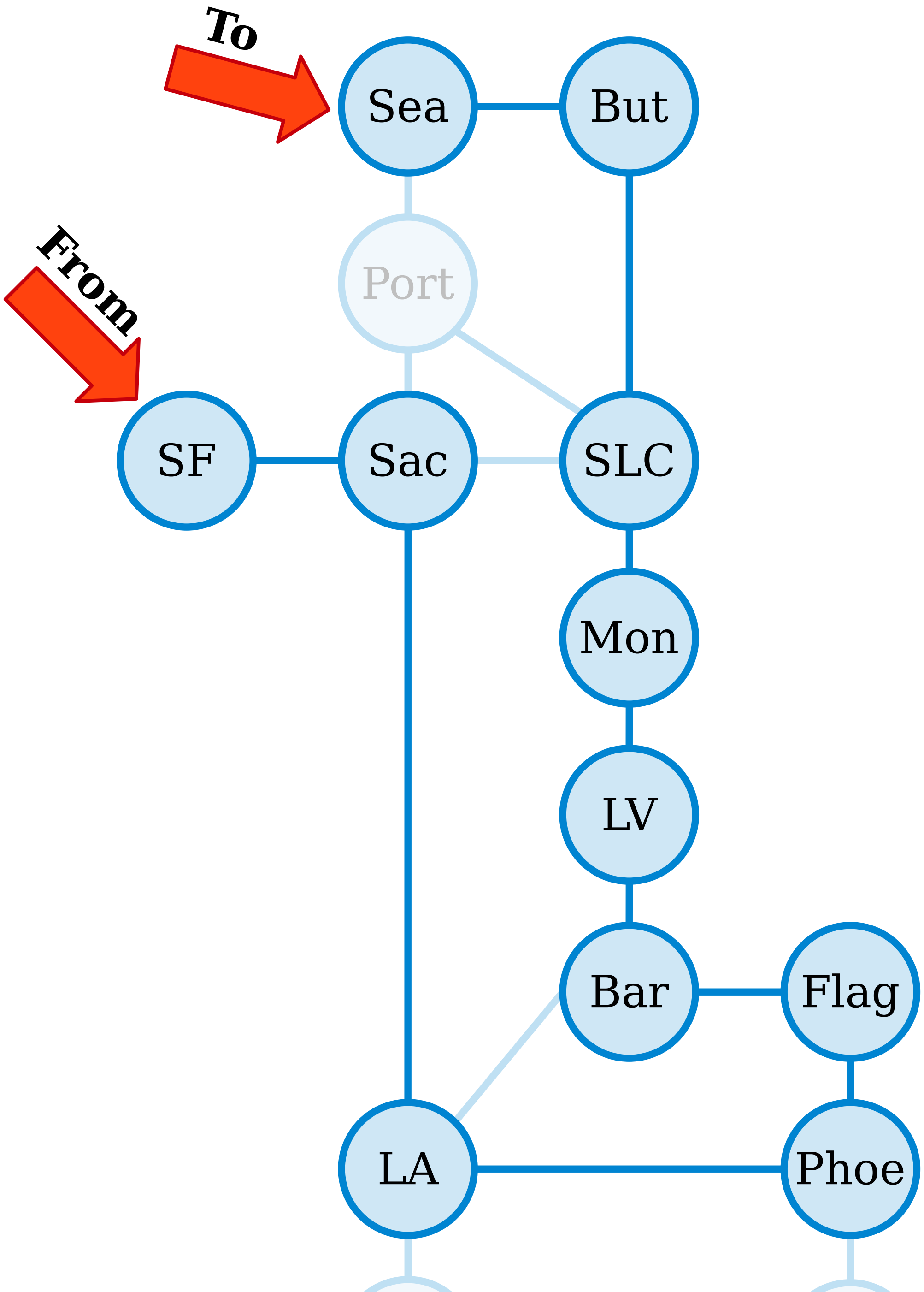

Sea
But
Port
SF
Sac
SLC
Mon
LV
Bar
Flag
LA
Phoe
SD
Nog



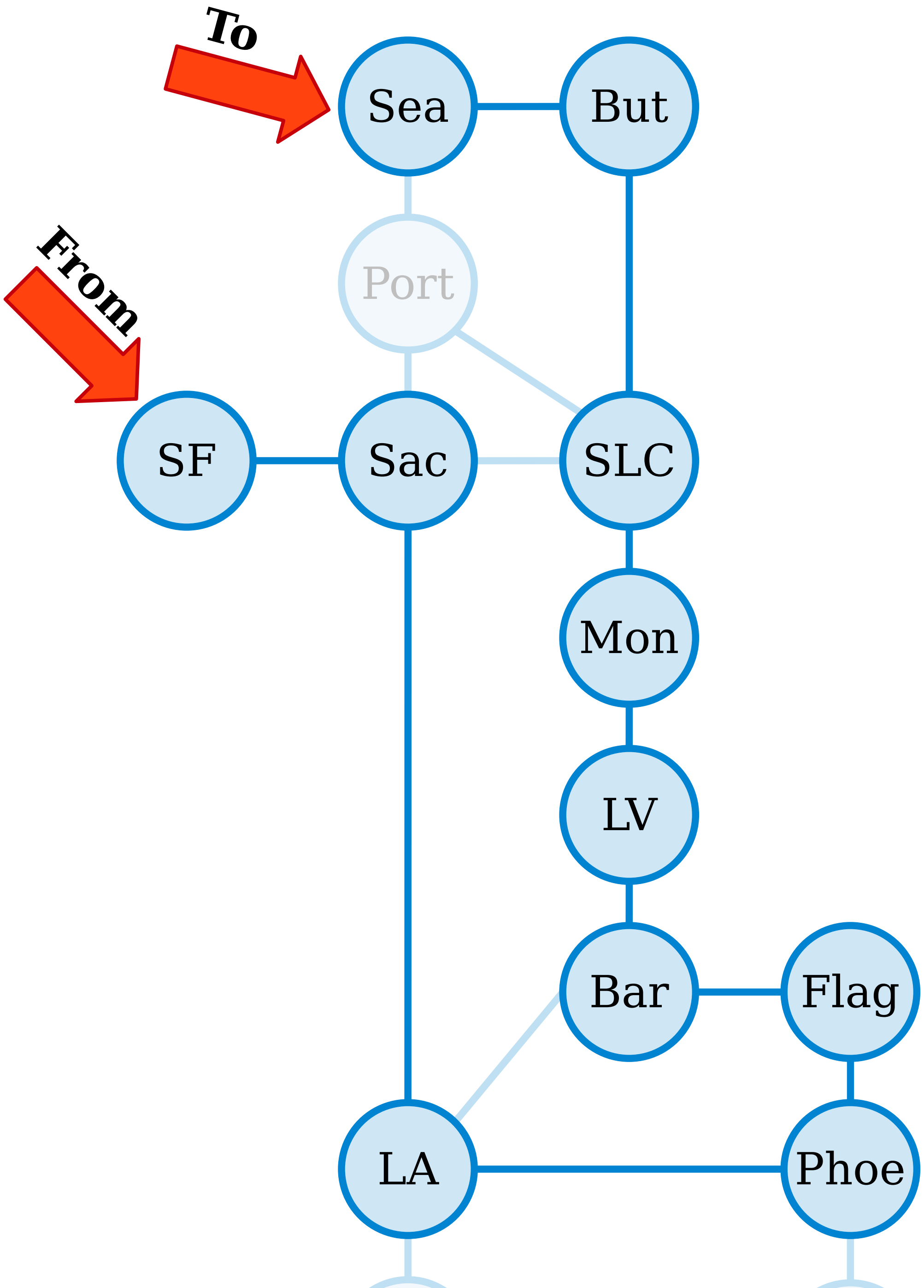
To
From
Sea
But
Port
SF
Sac
SLC
Mon
LV
Bar
Flag
LA
Phoe
SD
Nog


To
From
Sea
But
Port
SF
Sac
SLC
SF, Sac, Port, Sea
Mon
LV
Bar
Flag
LA
Phoe
SD
Nog


To
From
Sea
But
Port
SF
Sac
SLC
SF, Sac, SLC, Port, Sea
LV
Bar
Flag
LA
Phoe
SD
Nog

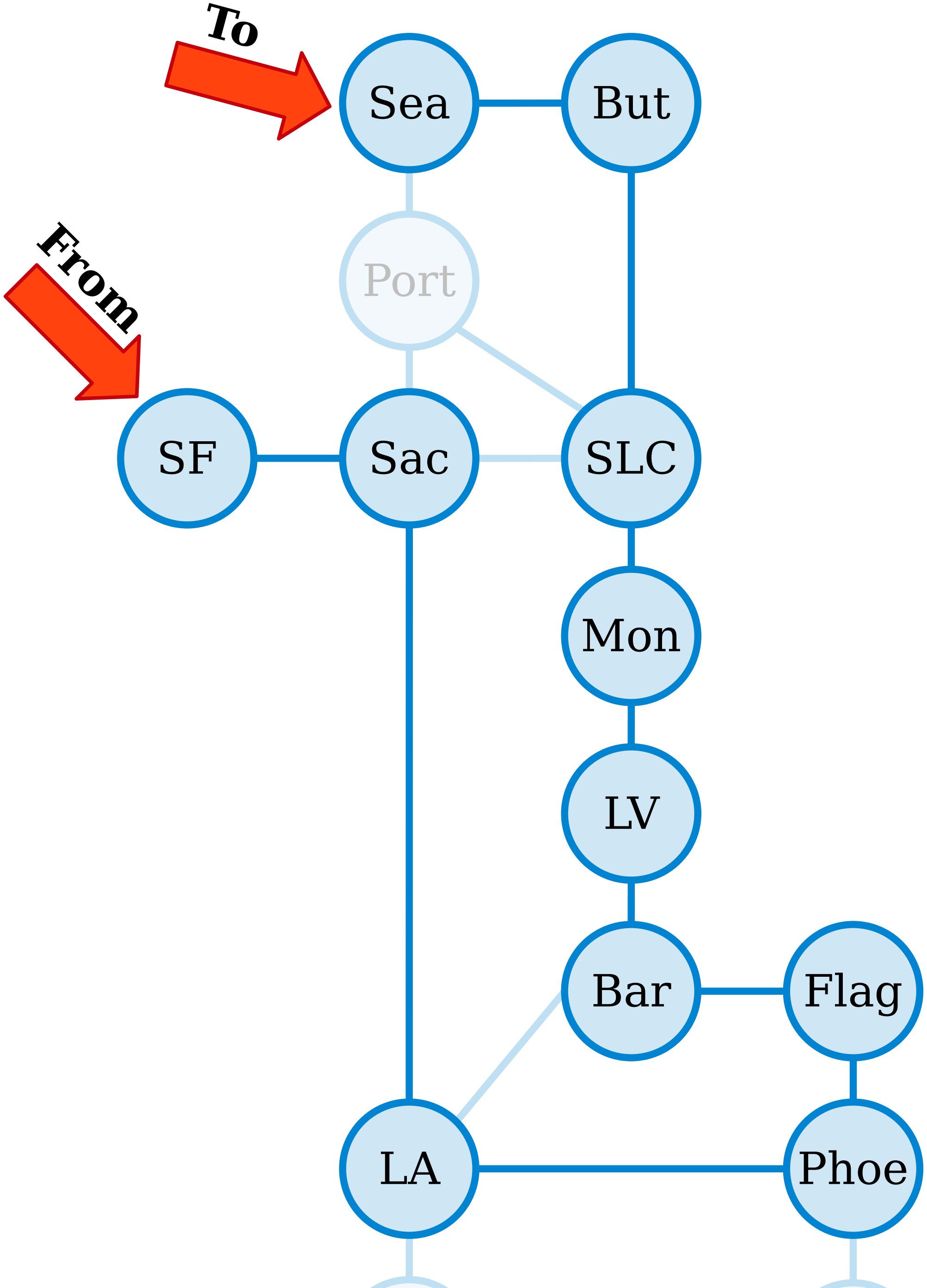


SF, Sac, LA, Phoe, Flag, Bar, LV, Mon, SLC, But, Sea



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

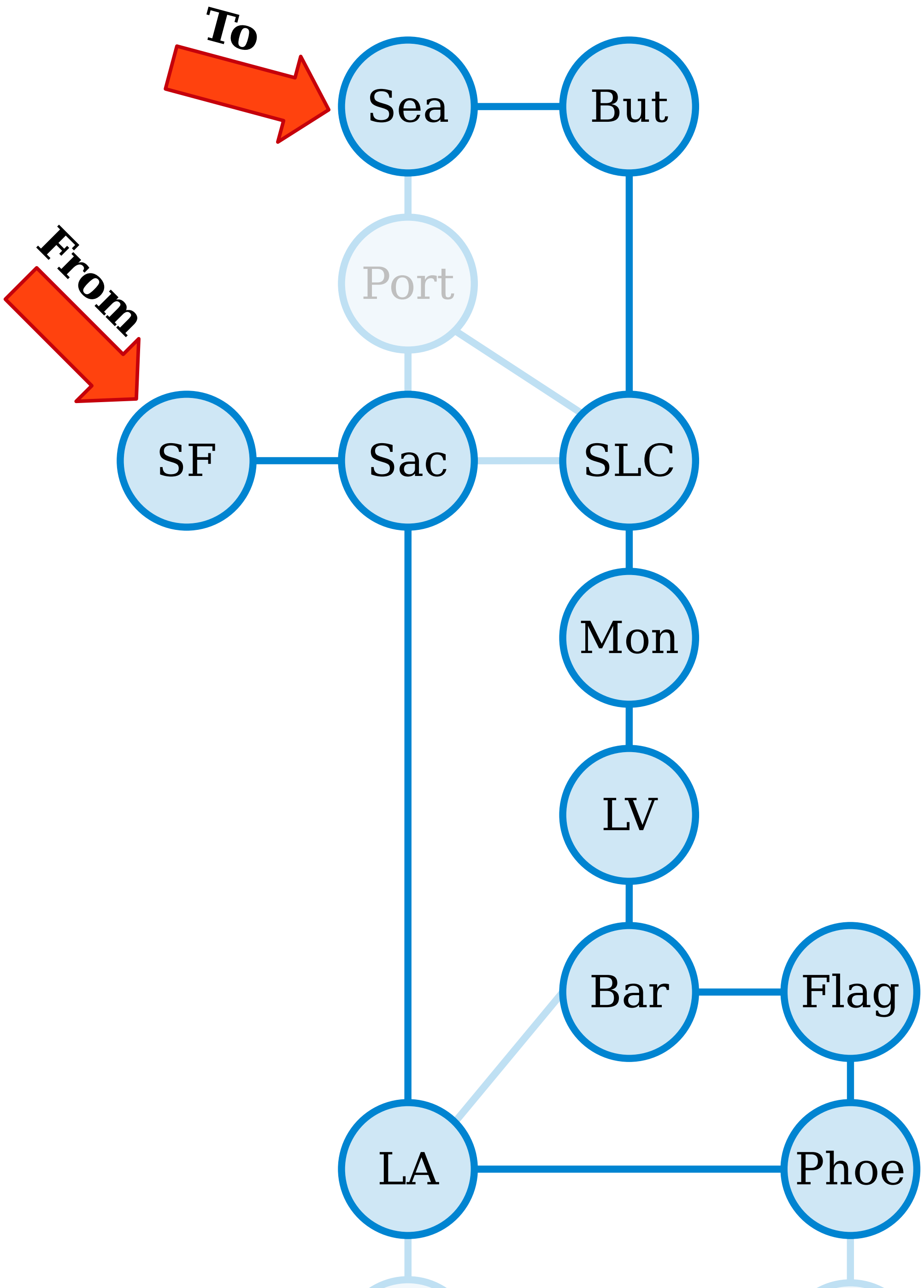
SF, Sac, LA, Phoe, Flag, Bar, LV, Mon, SLC, But, Sea



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

SF, Sac, LA, Phoe, Flag, Bar, LV, Mon, SLC, But, Sea

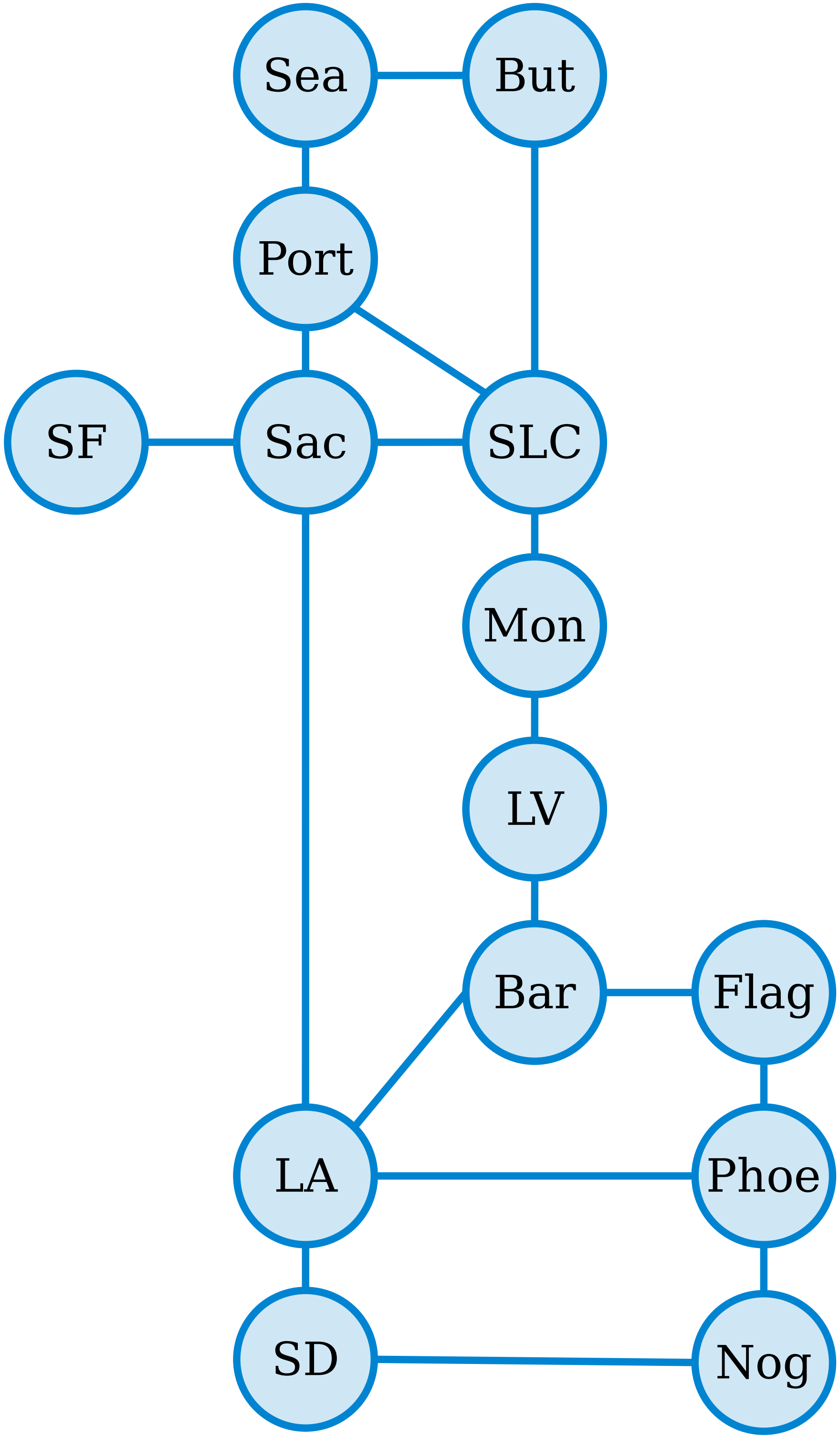


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

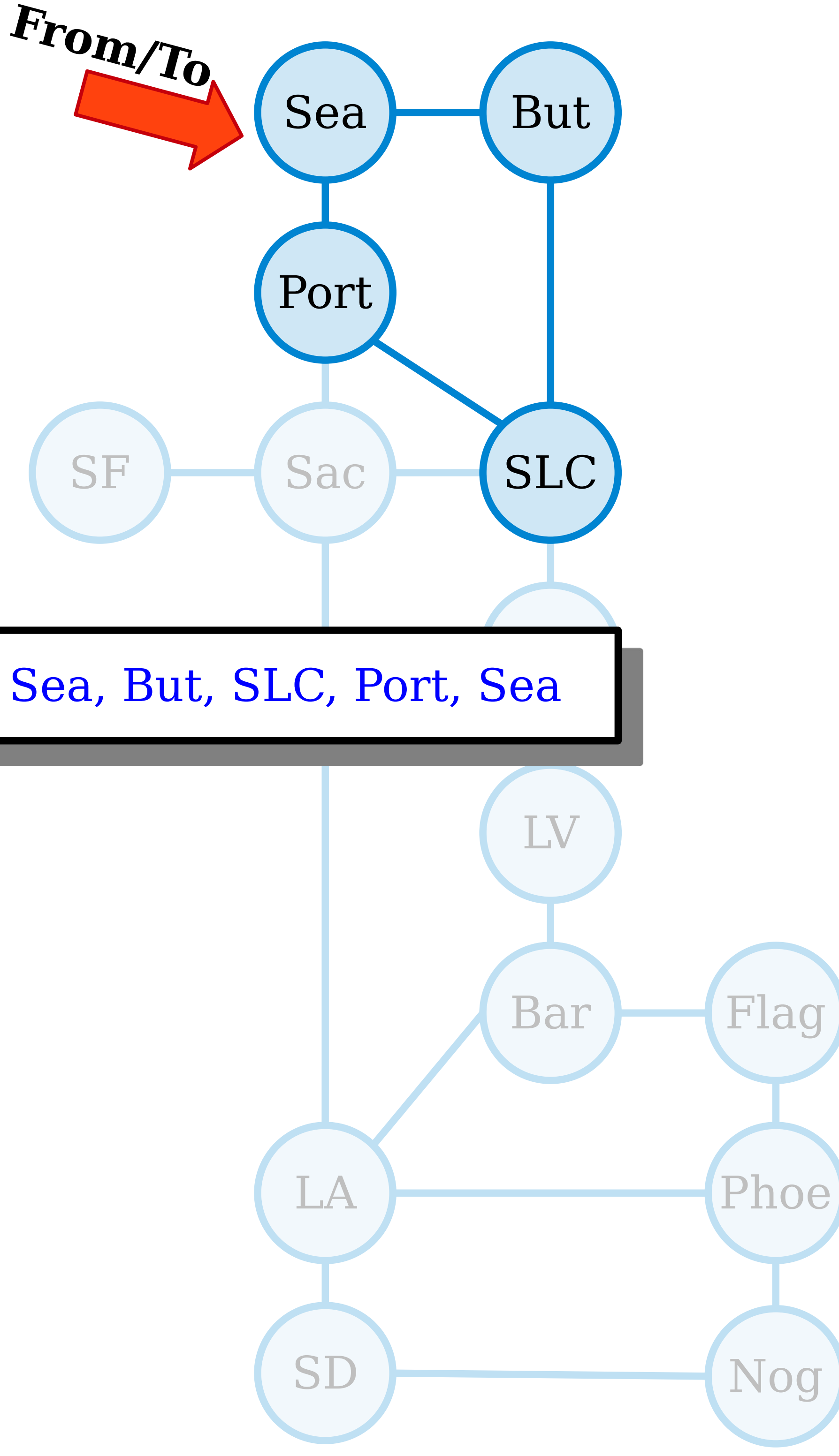
(This walk has length 10, but visits 11 cities.)

SF, Sac, LA, Phoe, Flag, Bar, LV, Mon, SLC, But, Sea



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

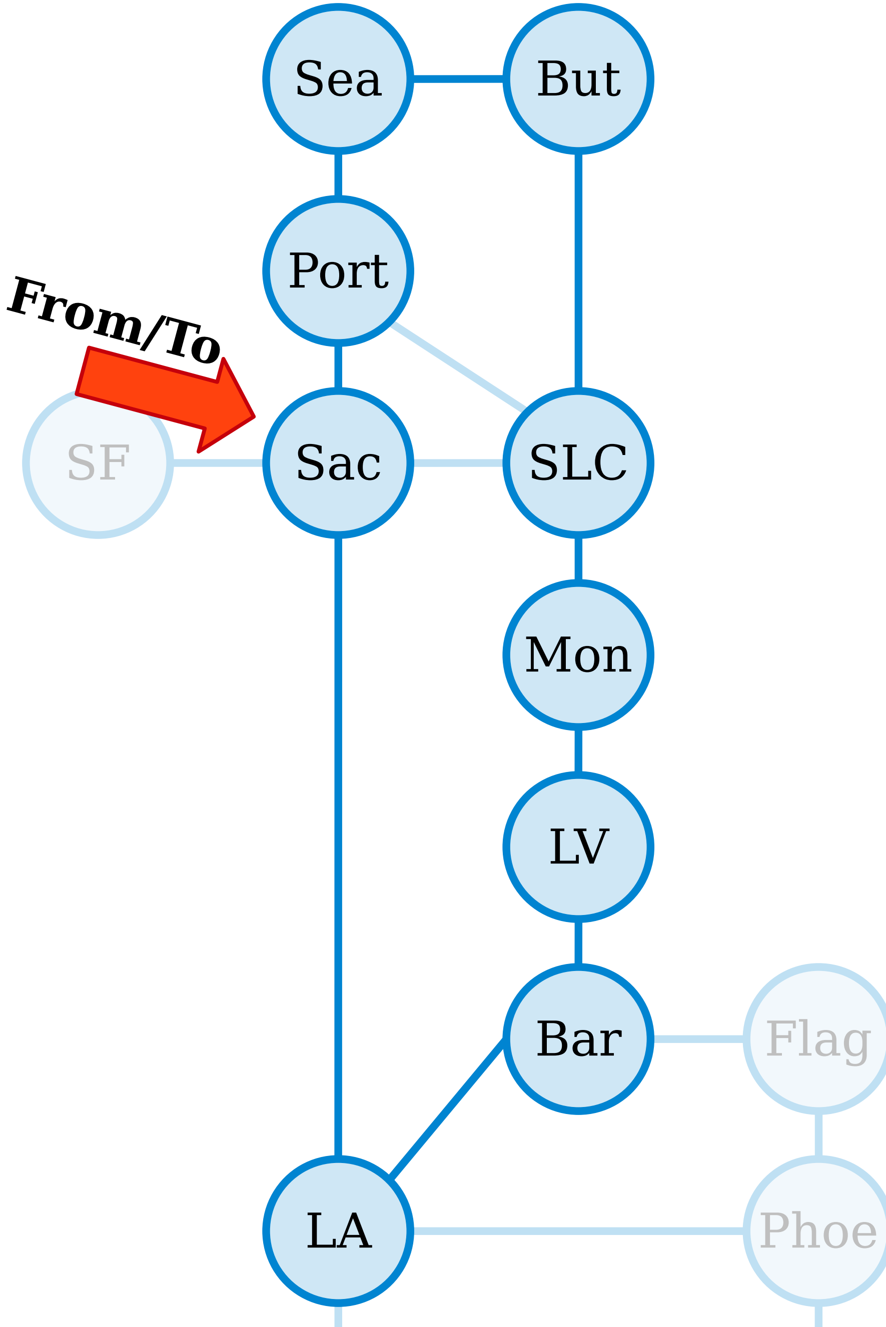
The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.



Sea, But, SLC, Port, Sea

A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

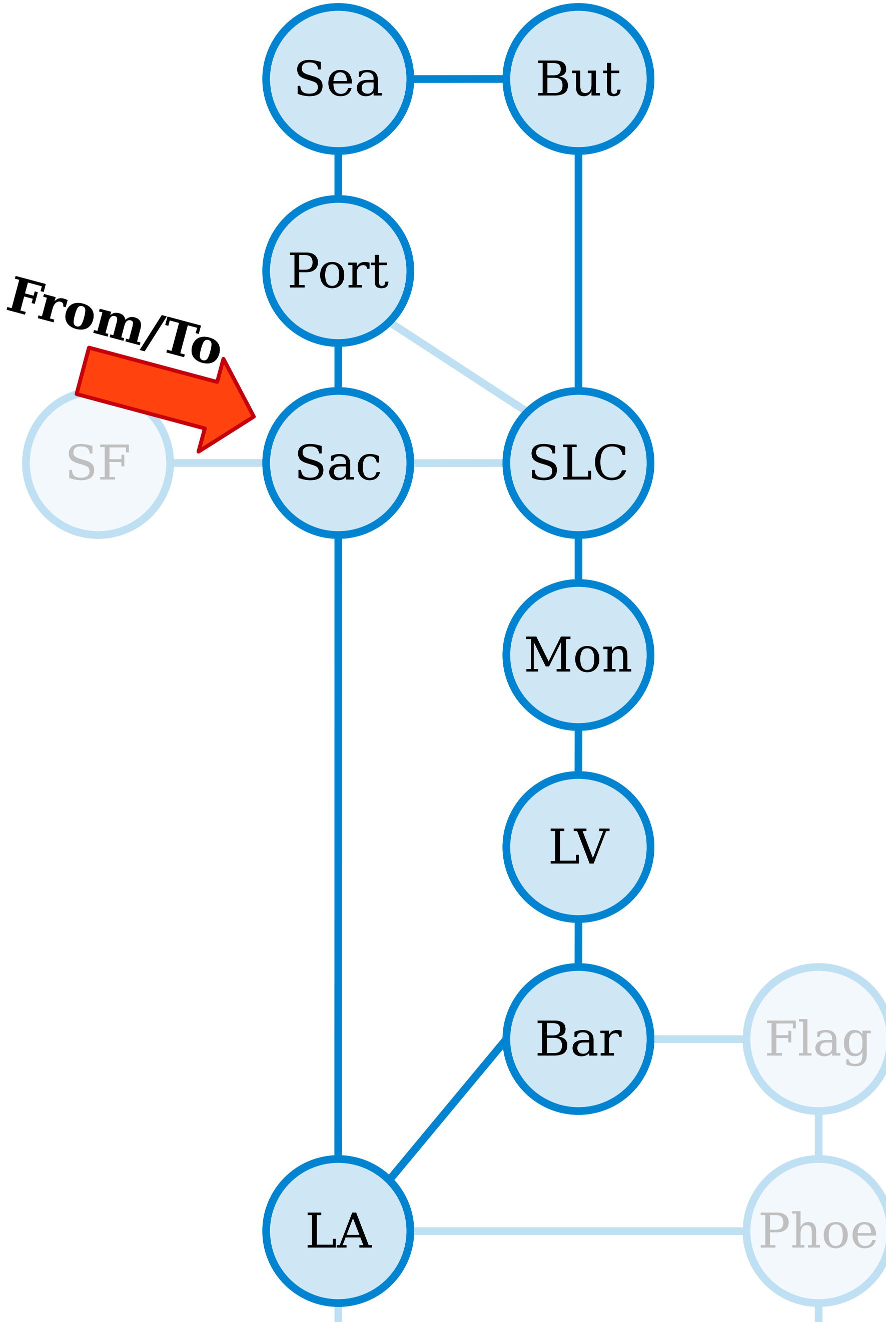
The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

Sac, Port, Sea, But, SLC, Mon, LV, Bar, LA, Sac

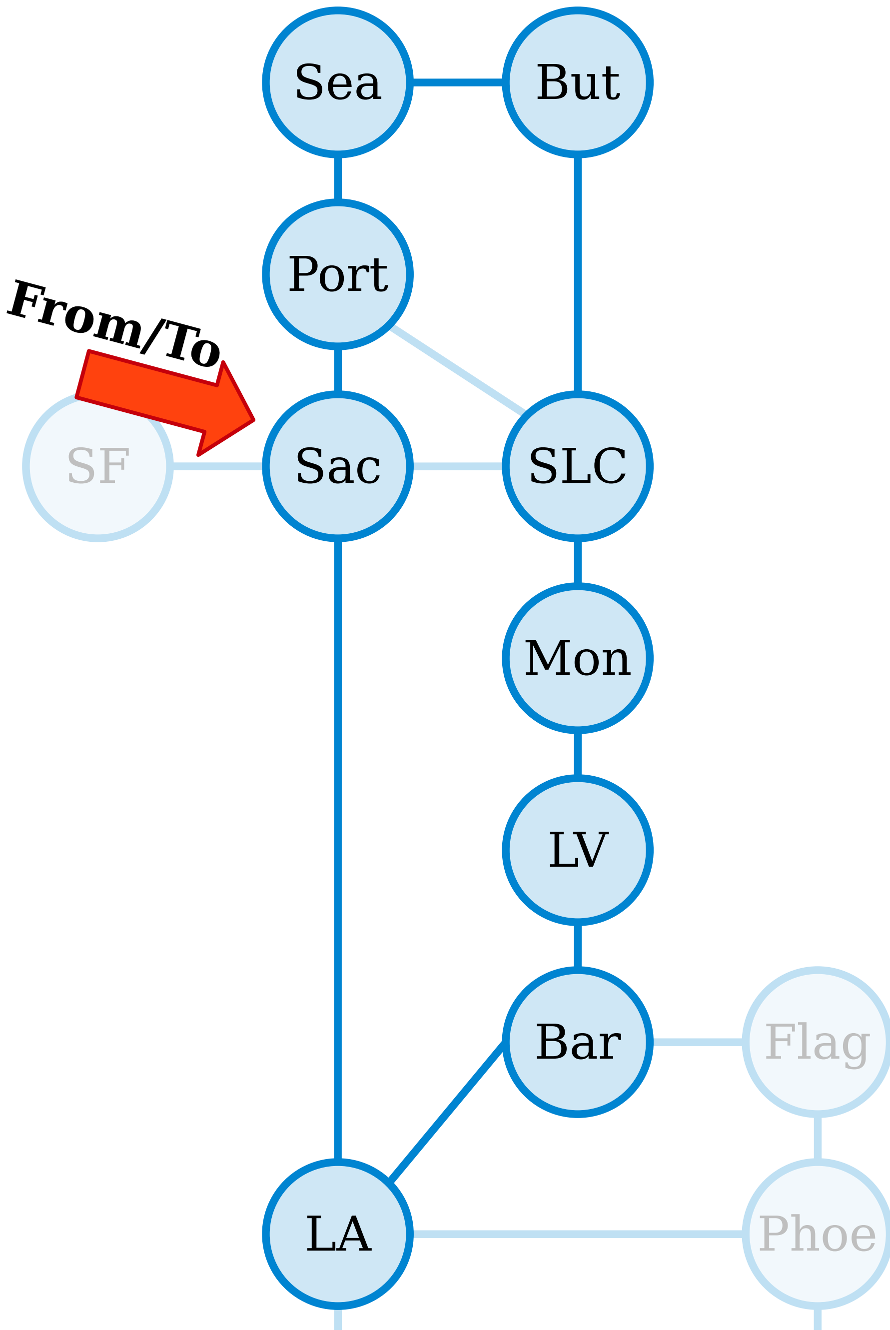


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

Sac, Port, Sea, But, SLC, Mon, LV, Bar, LA, Sac



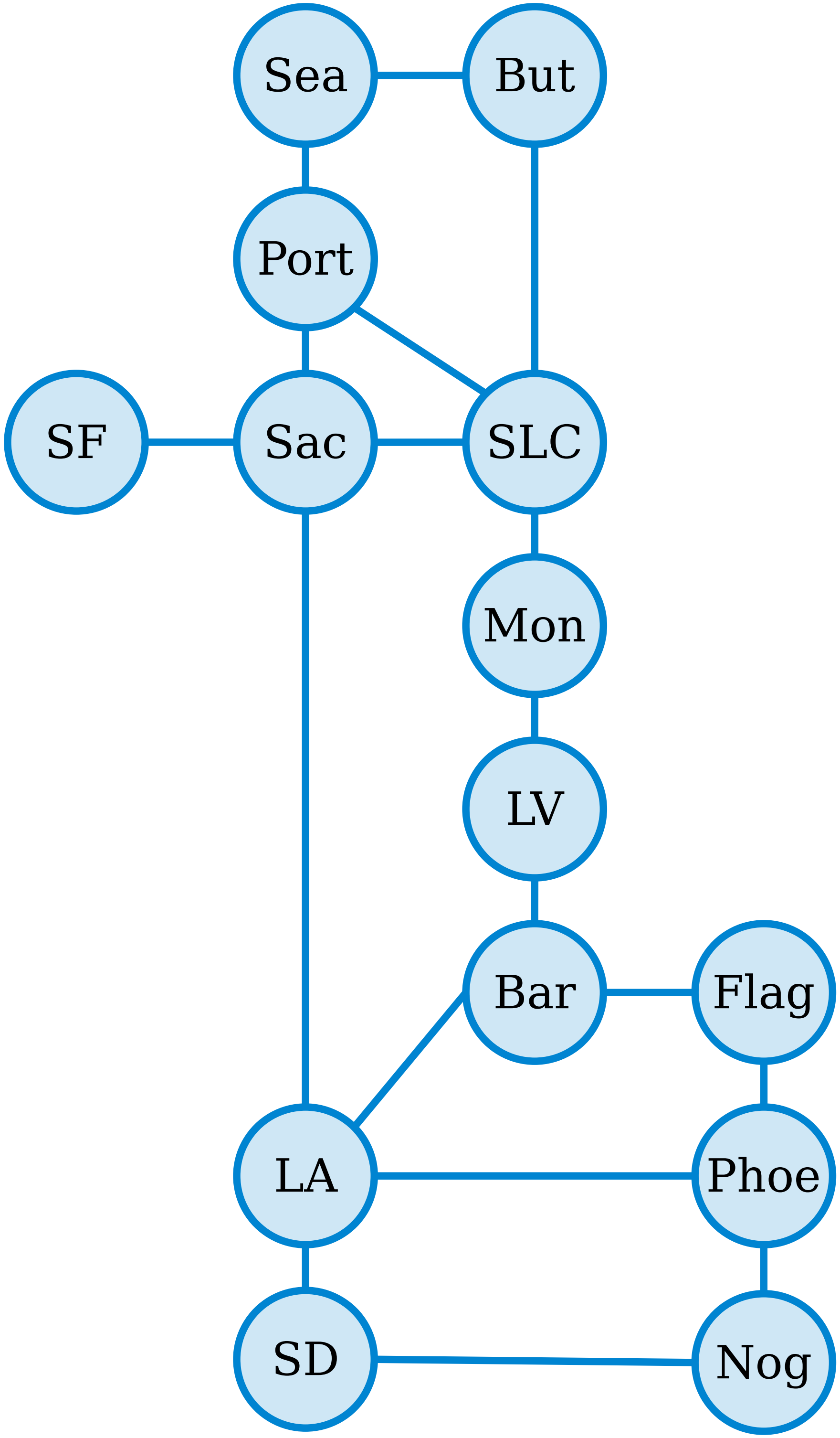
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

(This closed walk has length nine and visits nine different cities.)

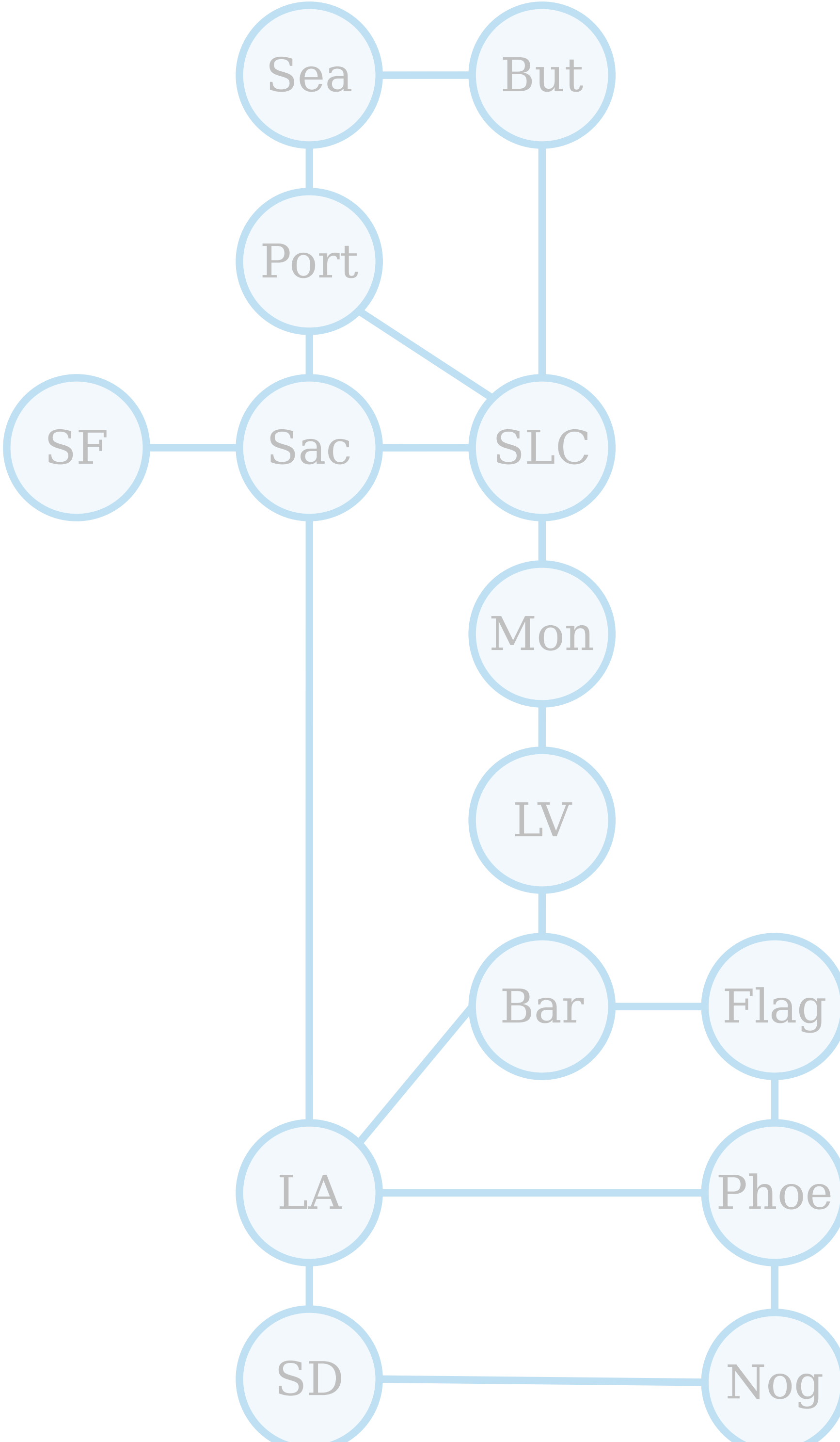
Sac, Port, Sea, But, SLC, Mon, LV, Bar, LA, Sac



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

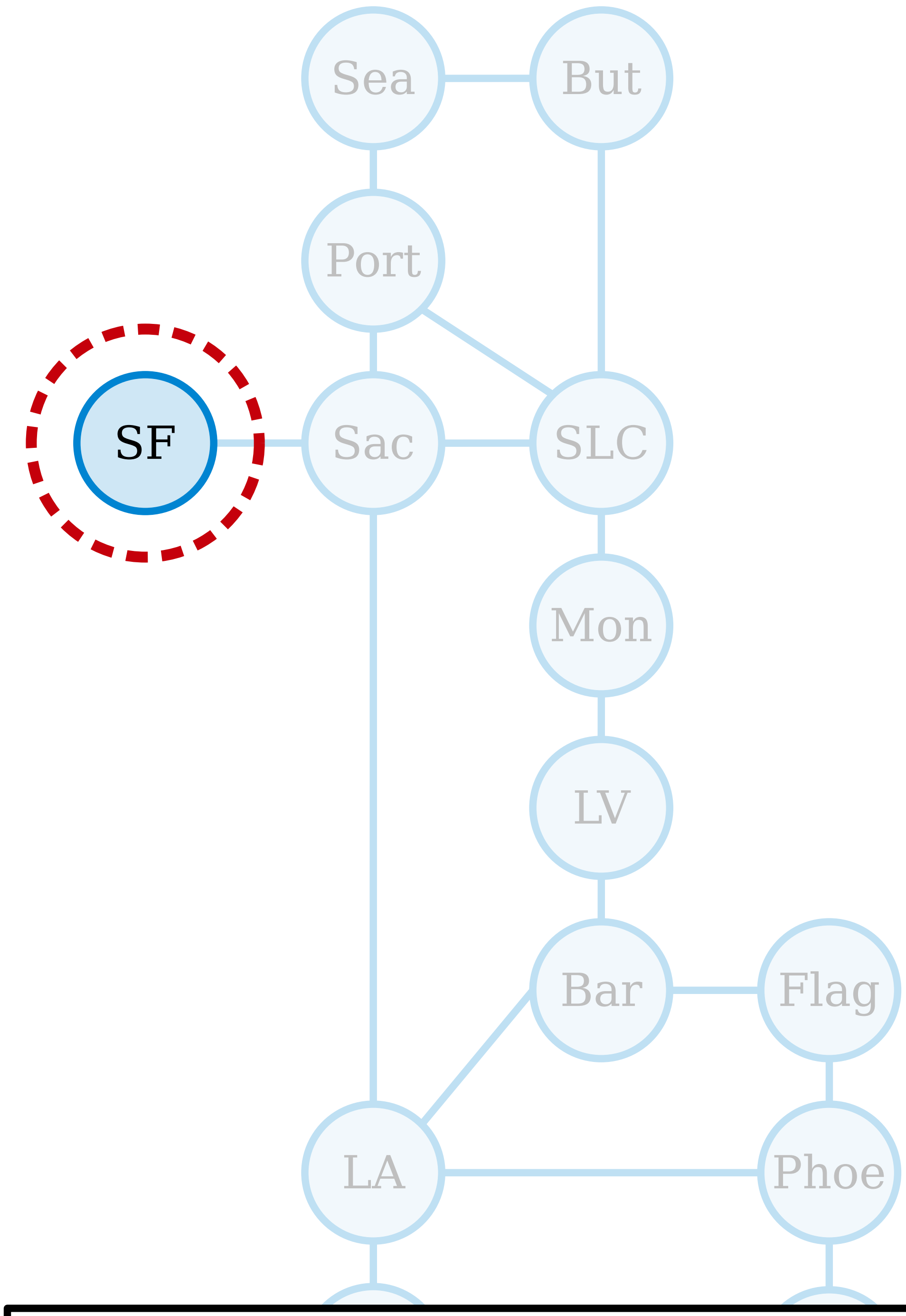
A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

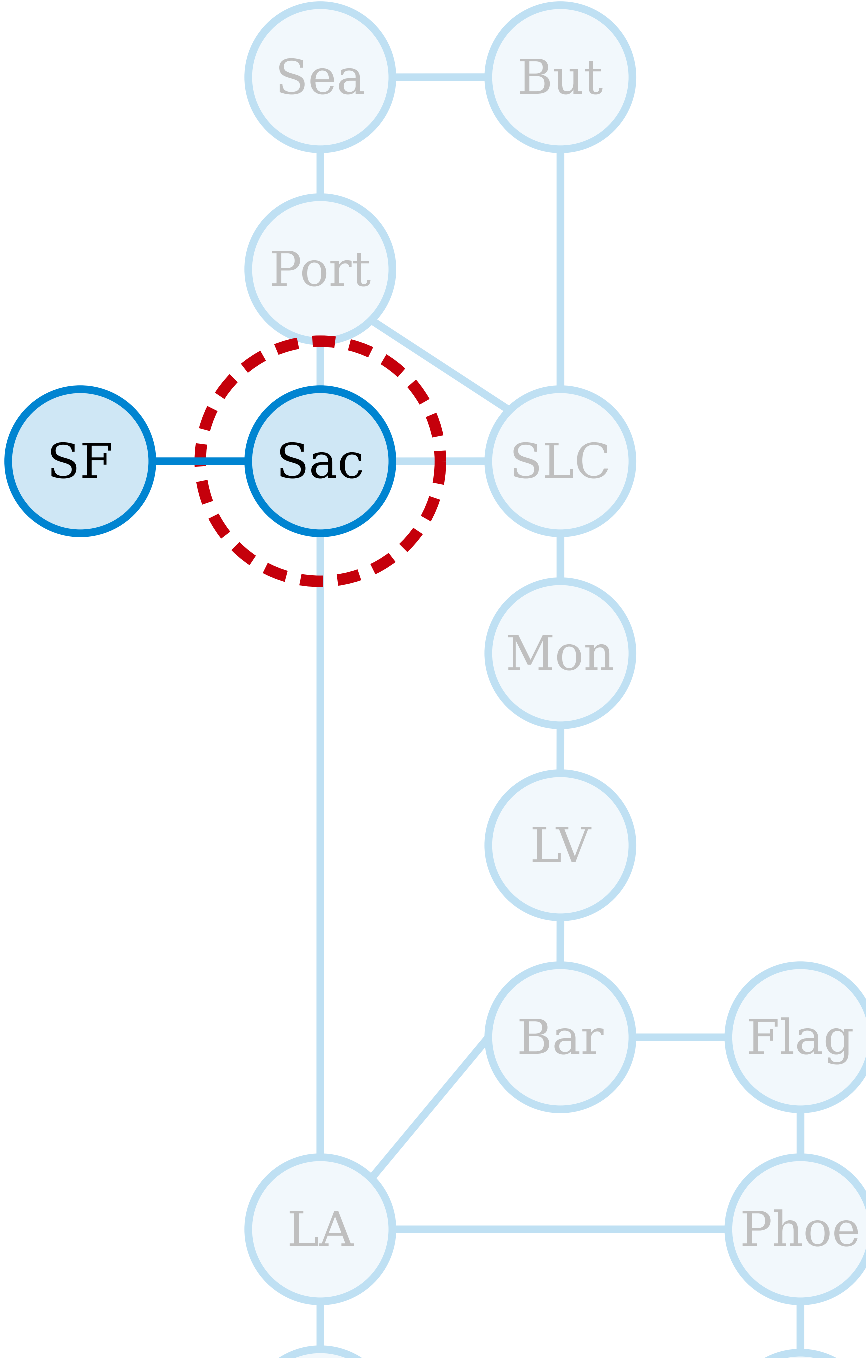


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

SF

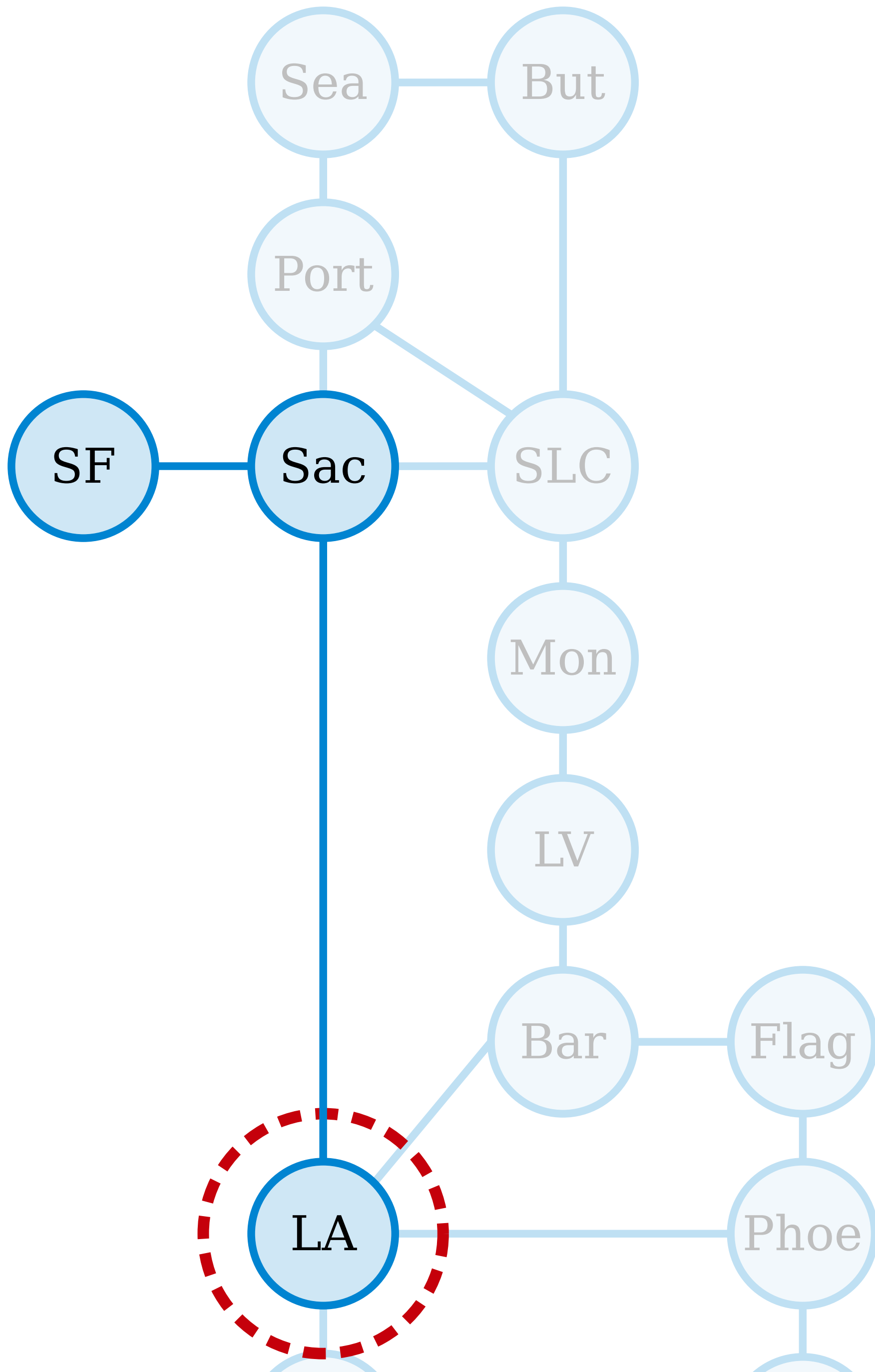


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

SF, Sac

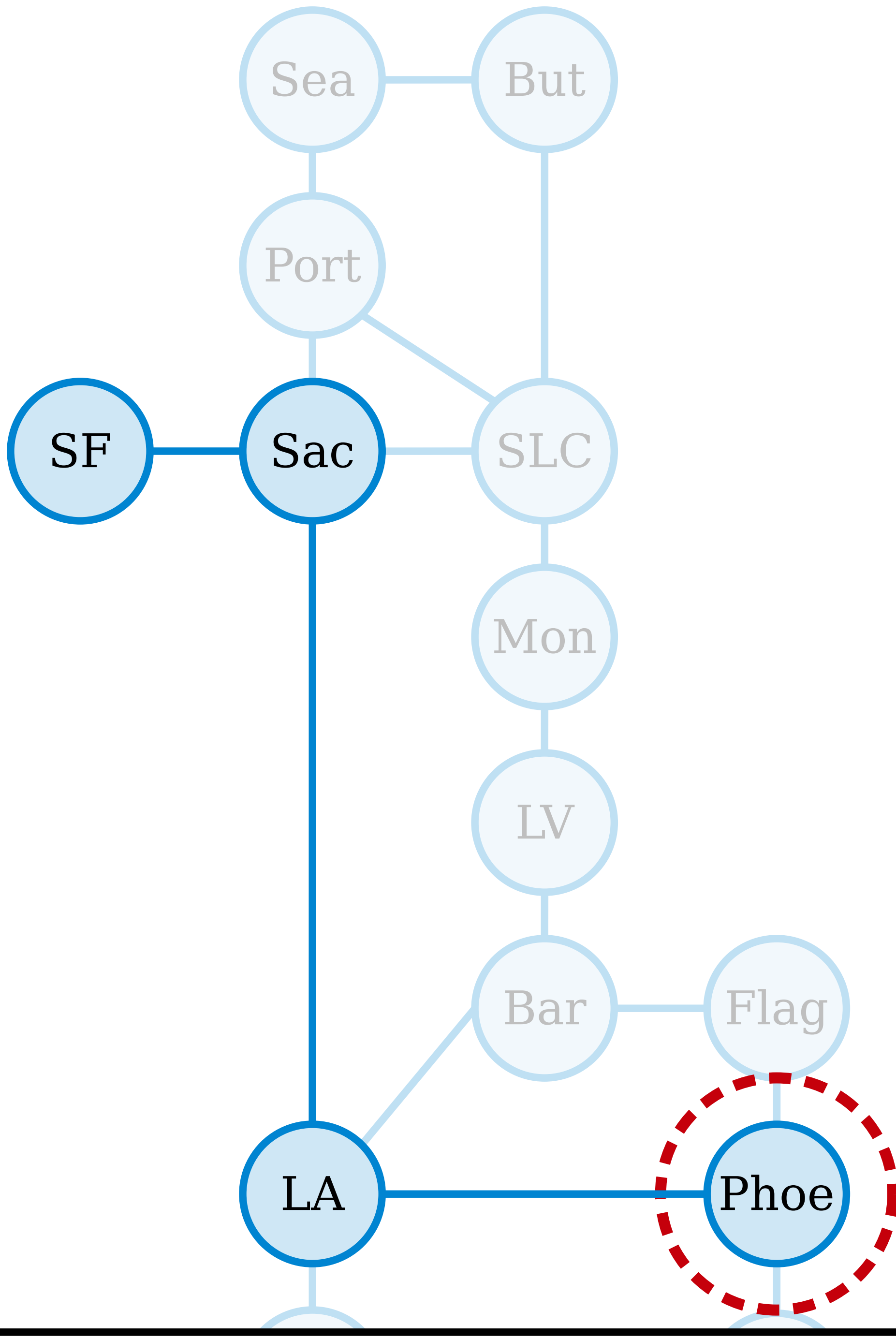


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

SF, Sac, LA

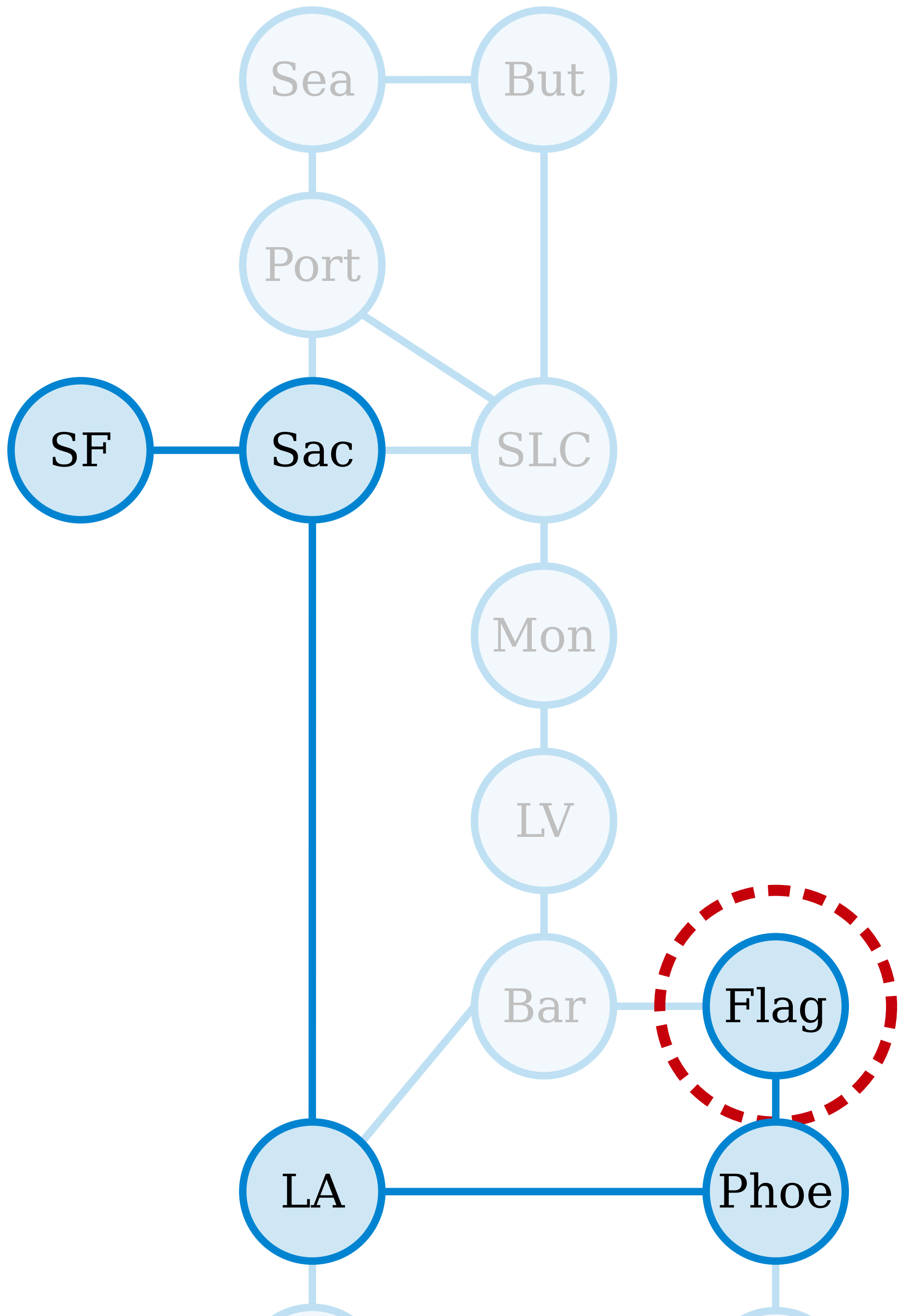


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

SF, Sac, LA, Phoe

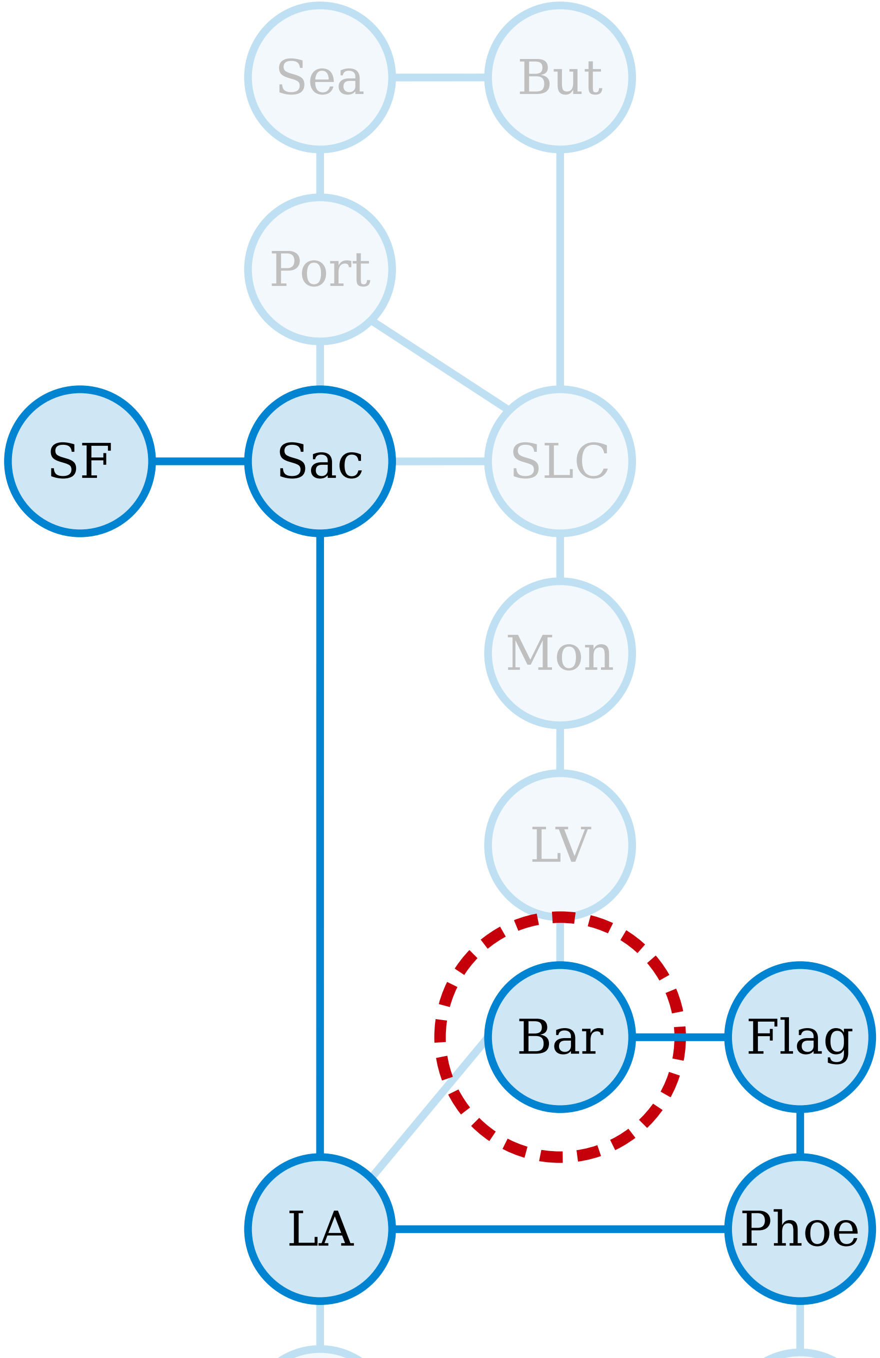


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

SF, Sac, LA, Phoe, Flag

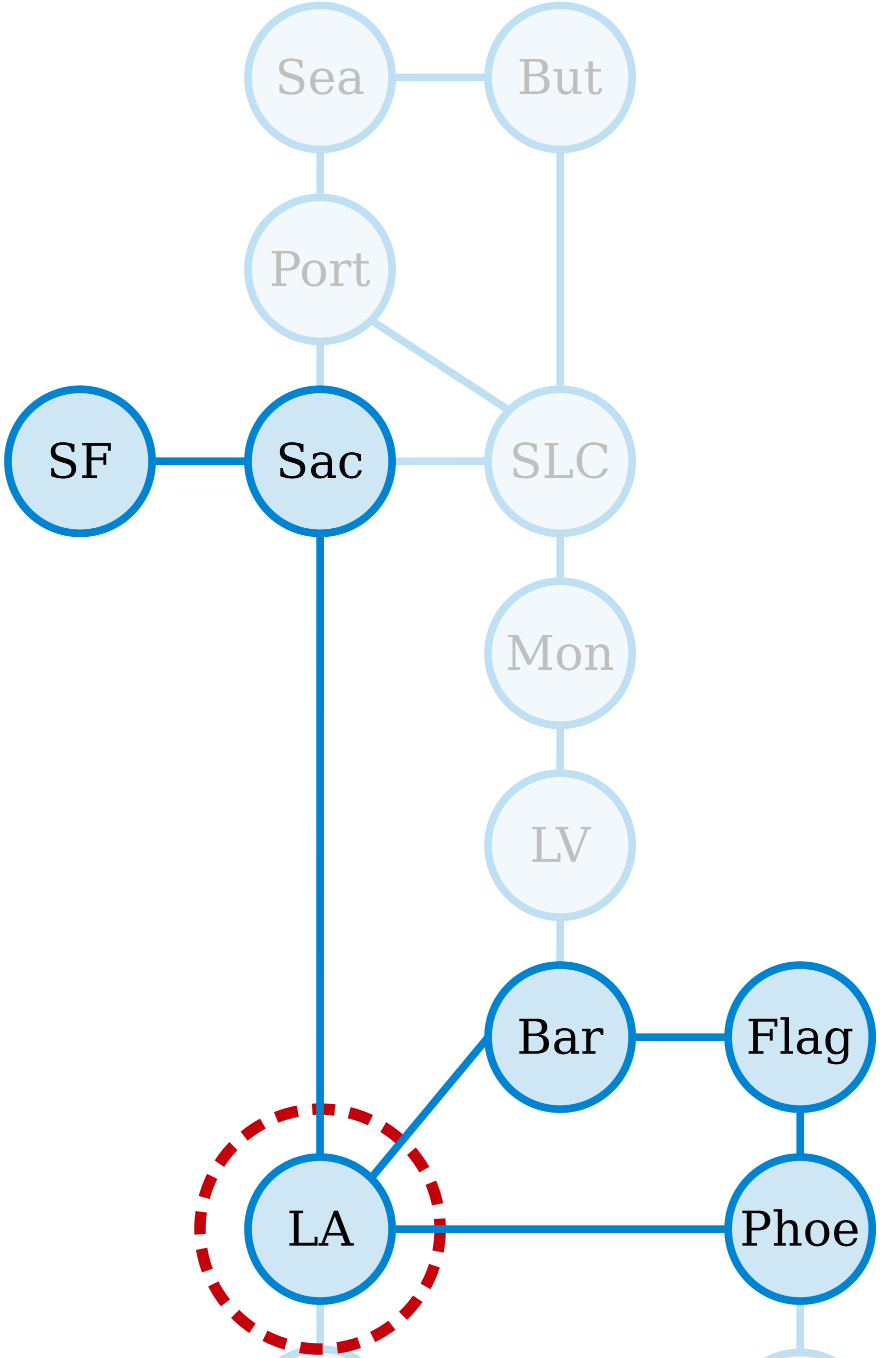


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

SF, Sac, LA, Phoe, Flag, Bar

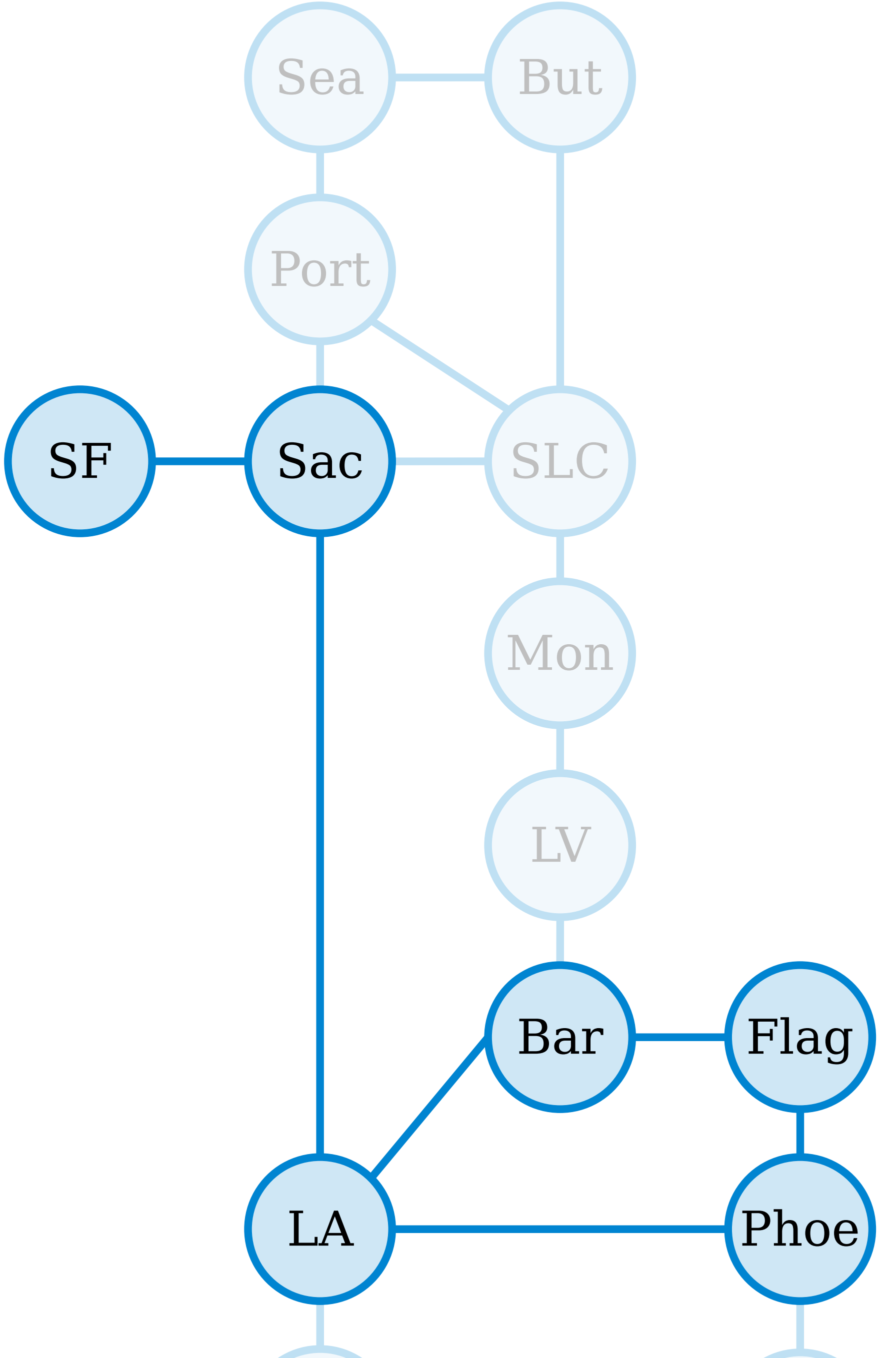


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

SF, Sac, LA, Phoe, Flag, Bar, LA

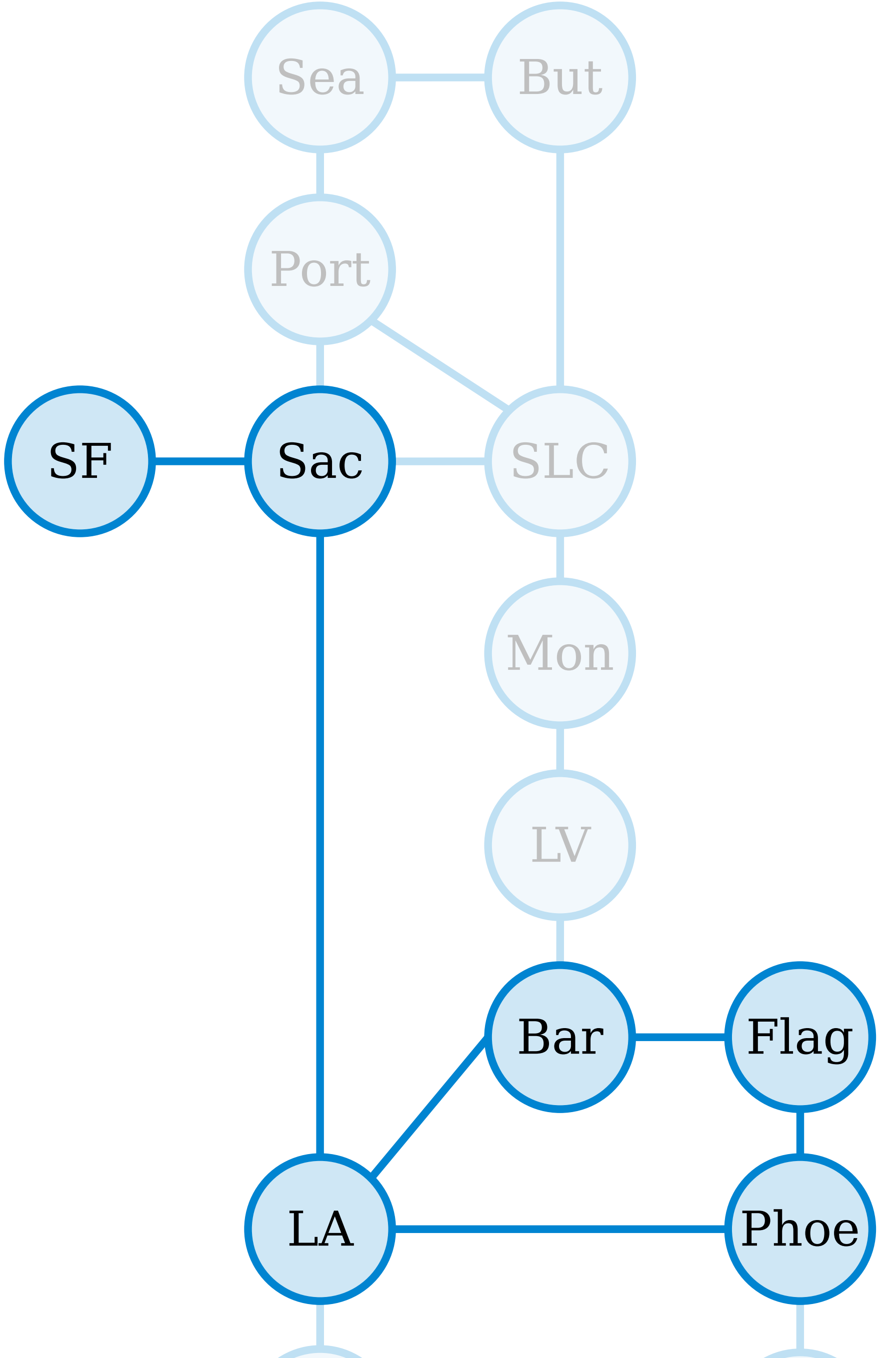


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

SF, Sac, LA, Phoe, Flag, Bar, LA



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.

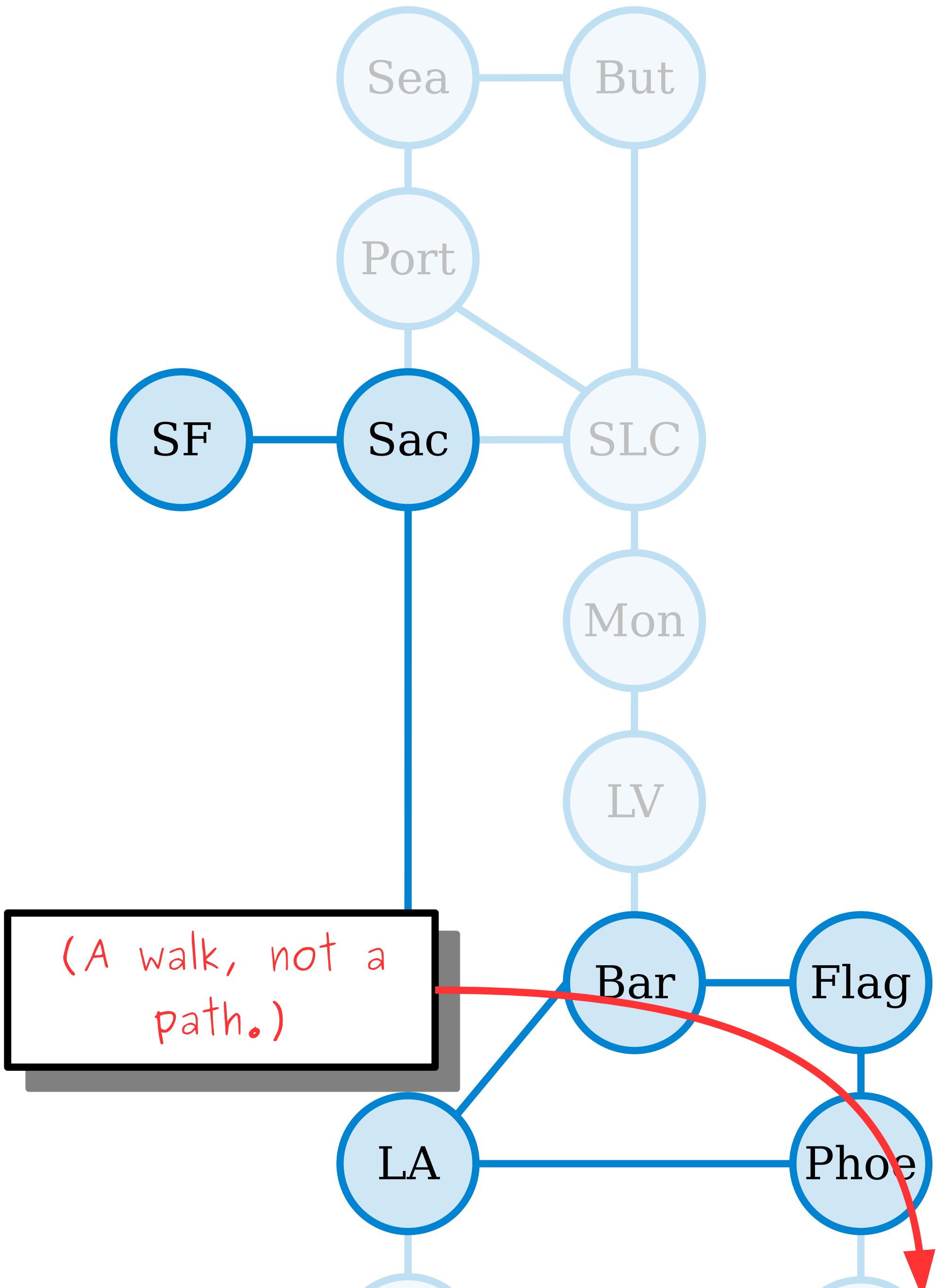
SF, Sac, LA, Phoe, Flag, Bar, LA

A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

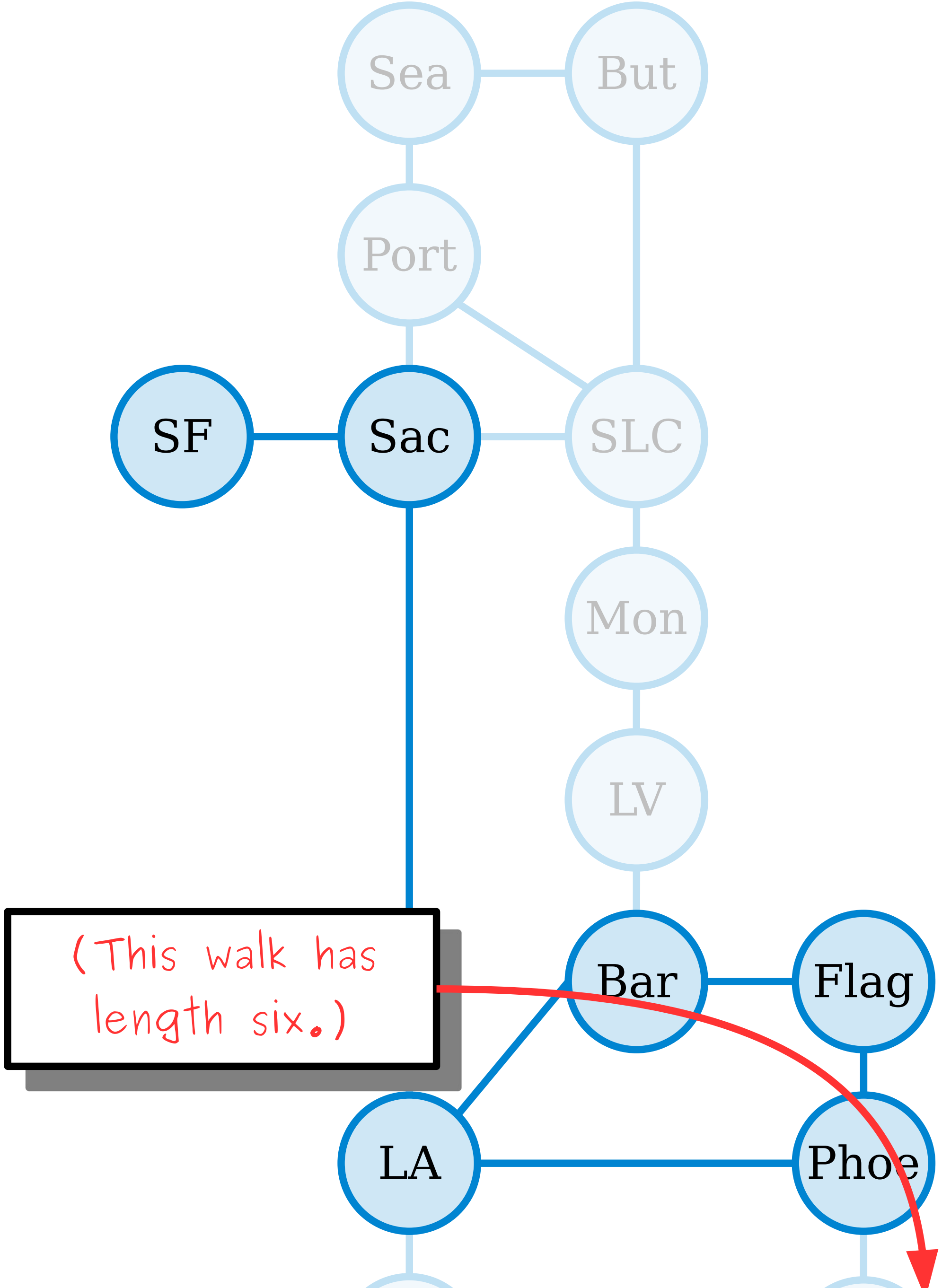
The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.



SF, Sac, LA, Phoe, Flag, Bar, LA



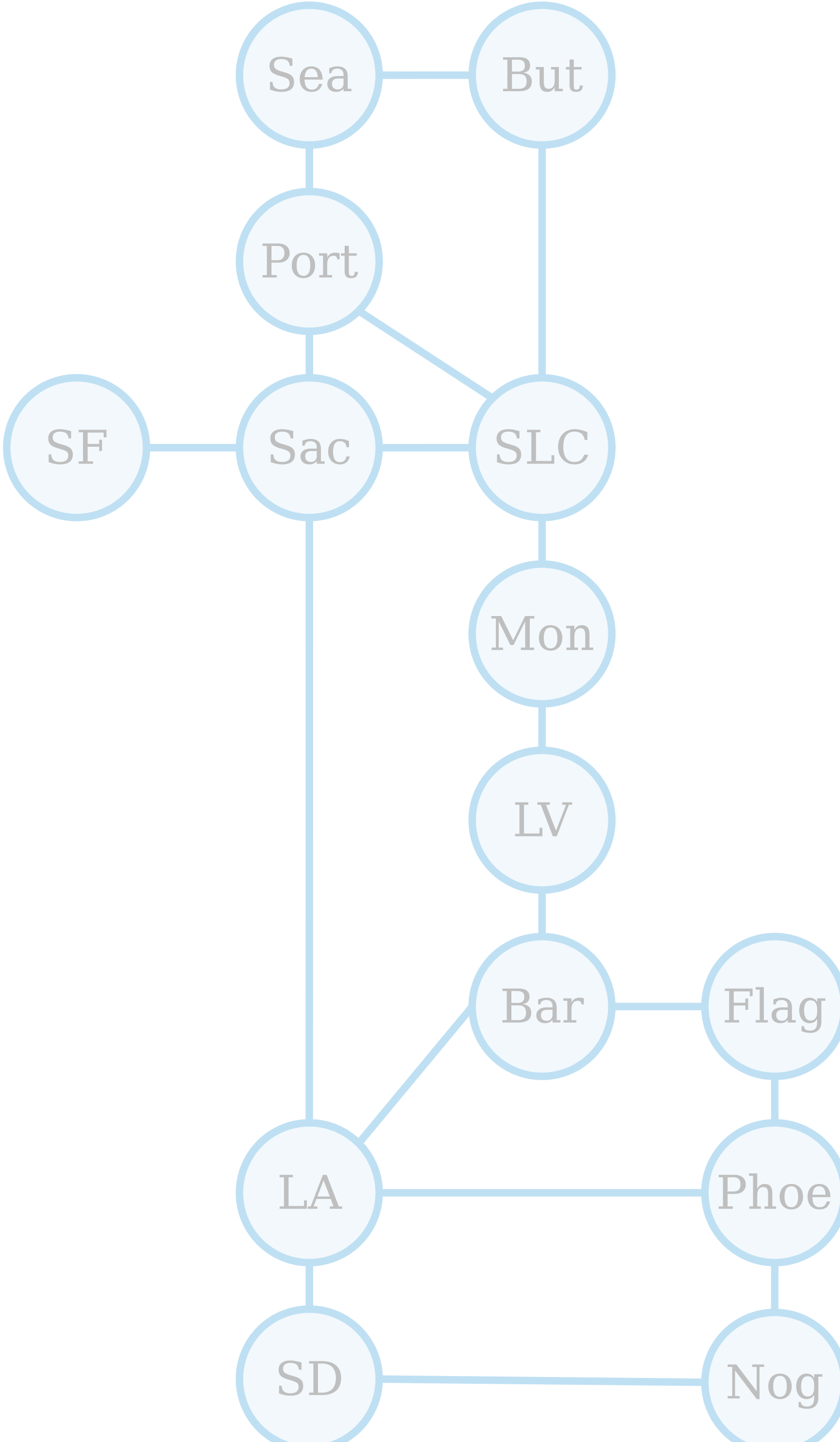
SF, Sac, LA, Phoe, Flag, Bar, LA

A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.

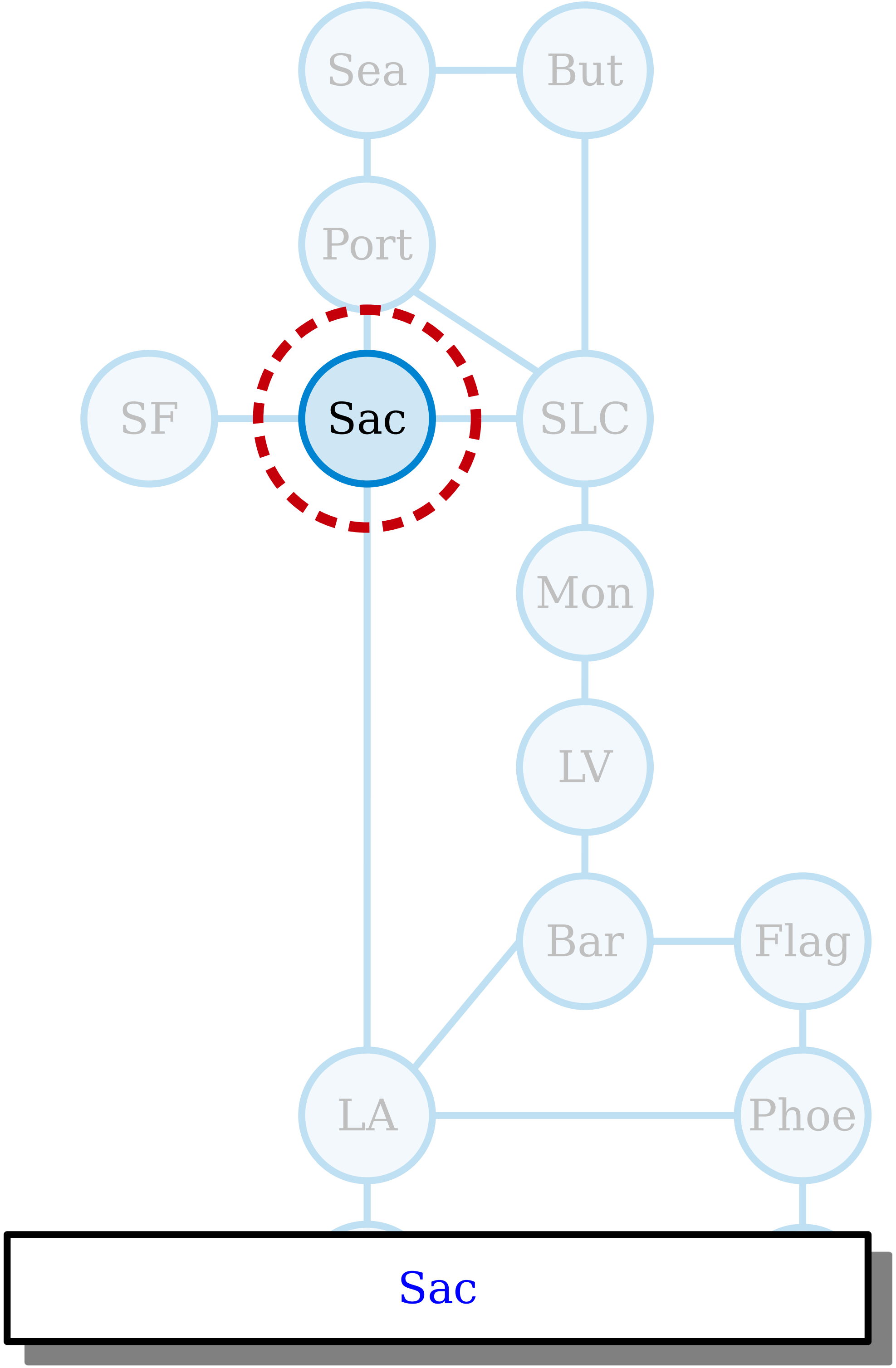


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.

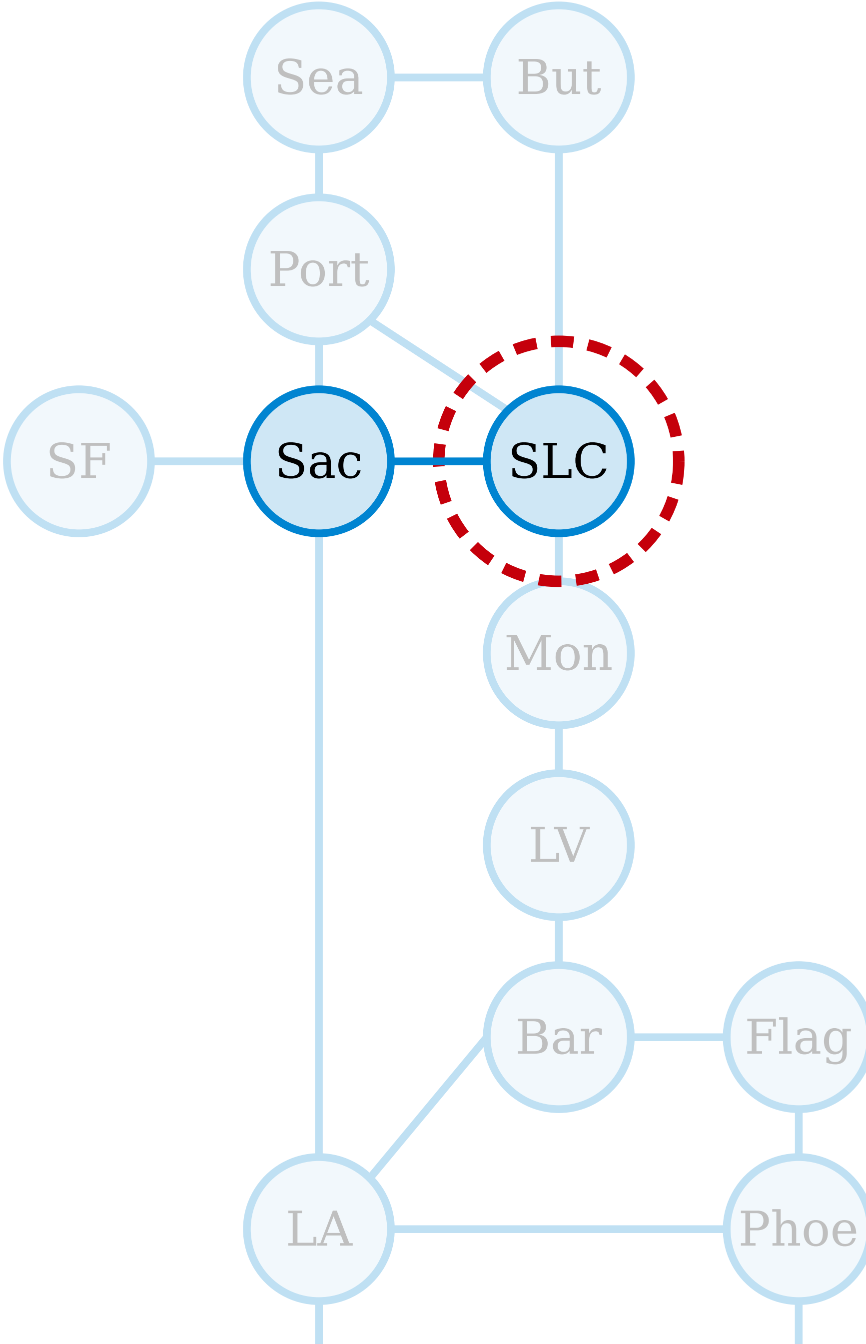


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.



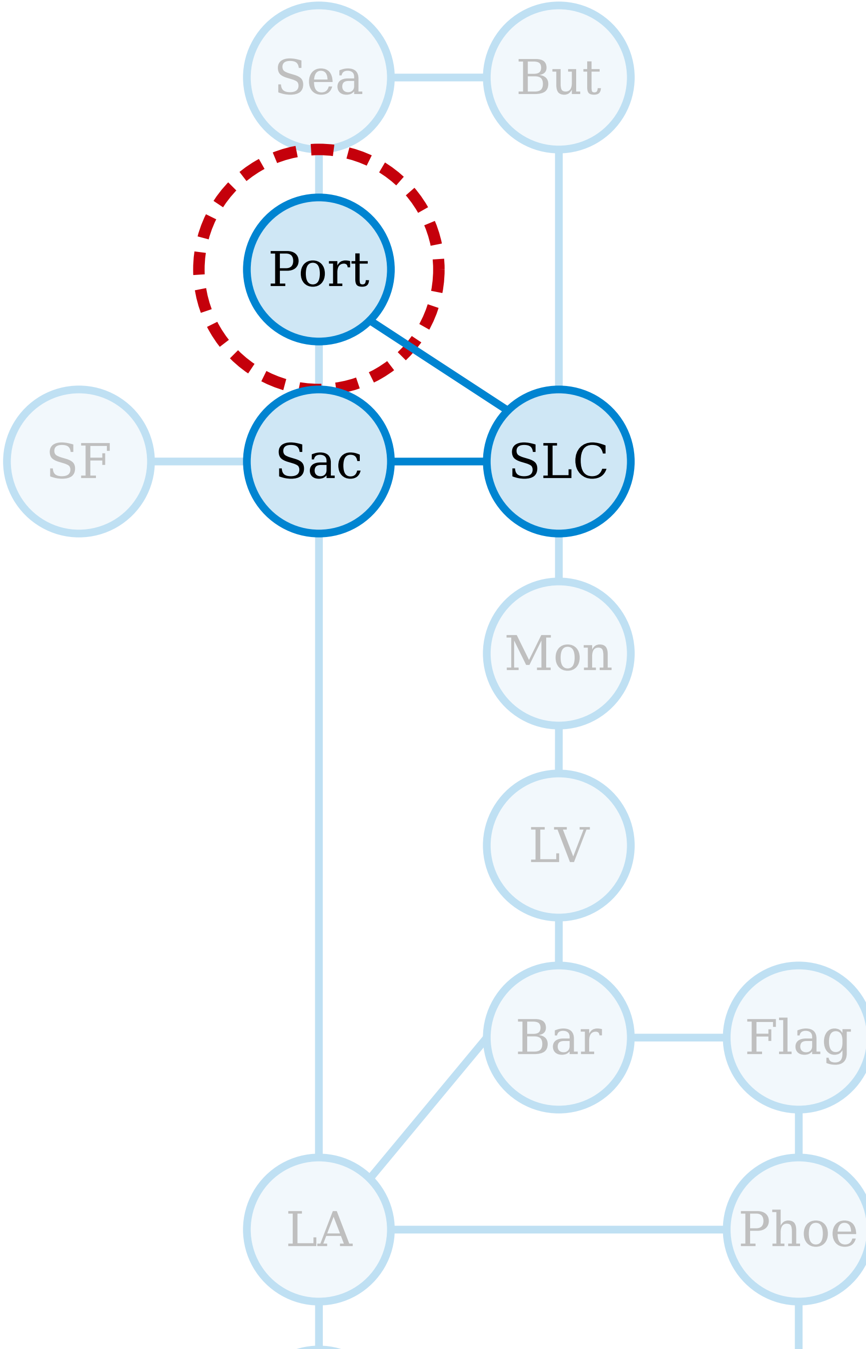
Sac, SLC

A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.



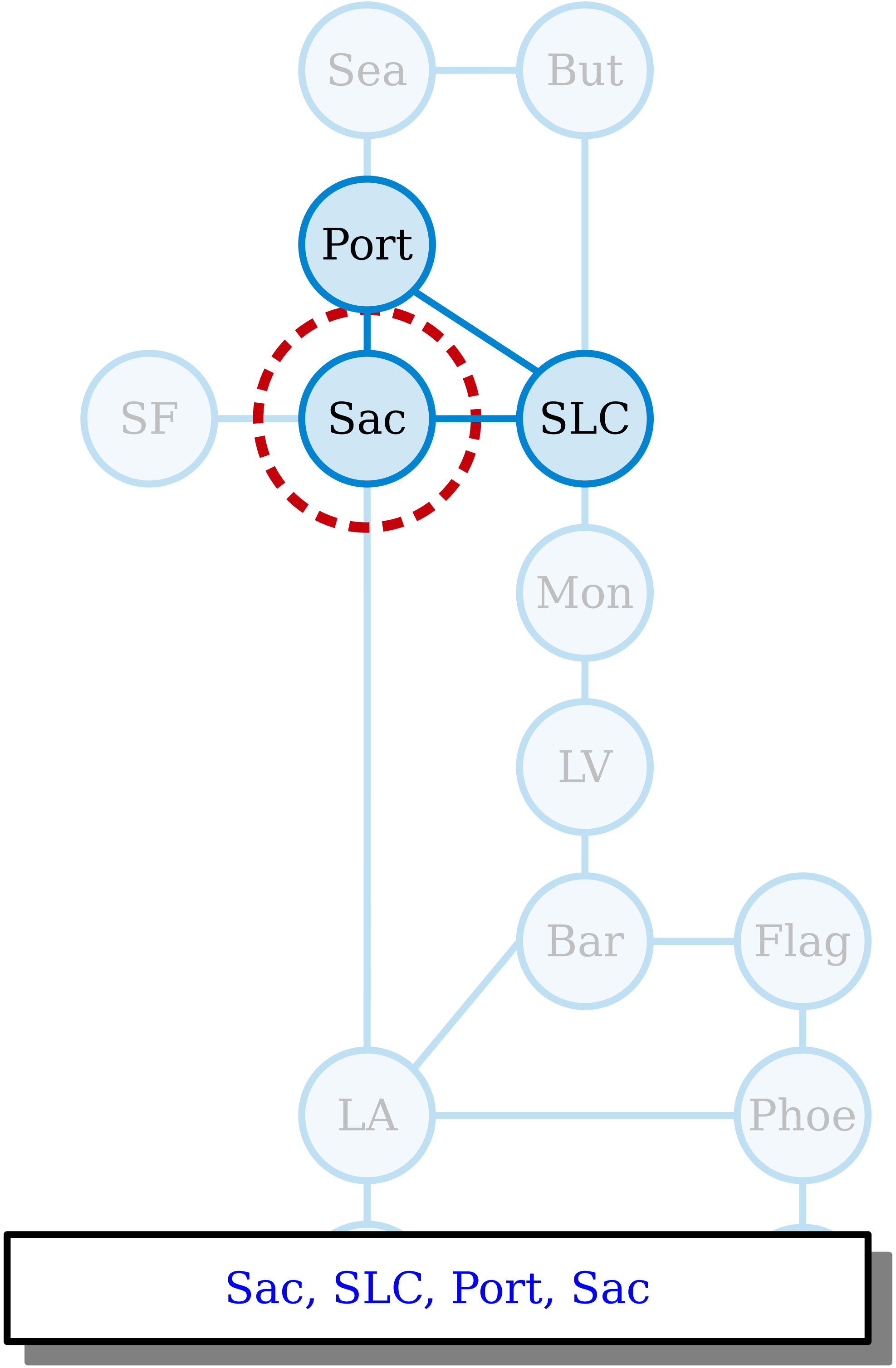
Sac, SLC, Port

A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.

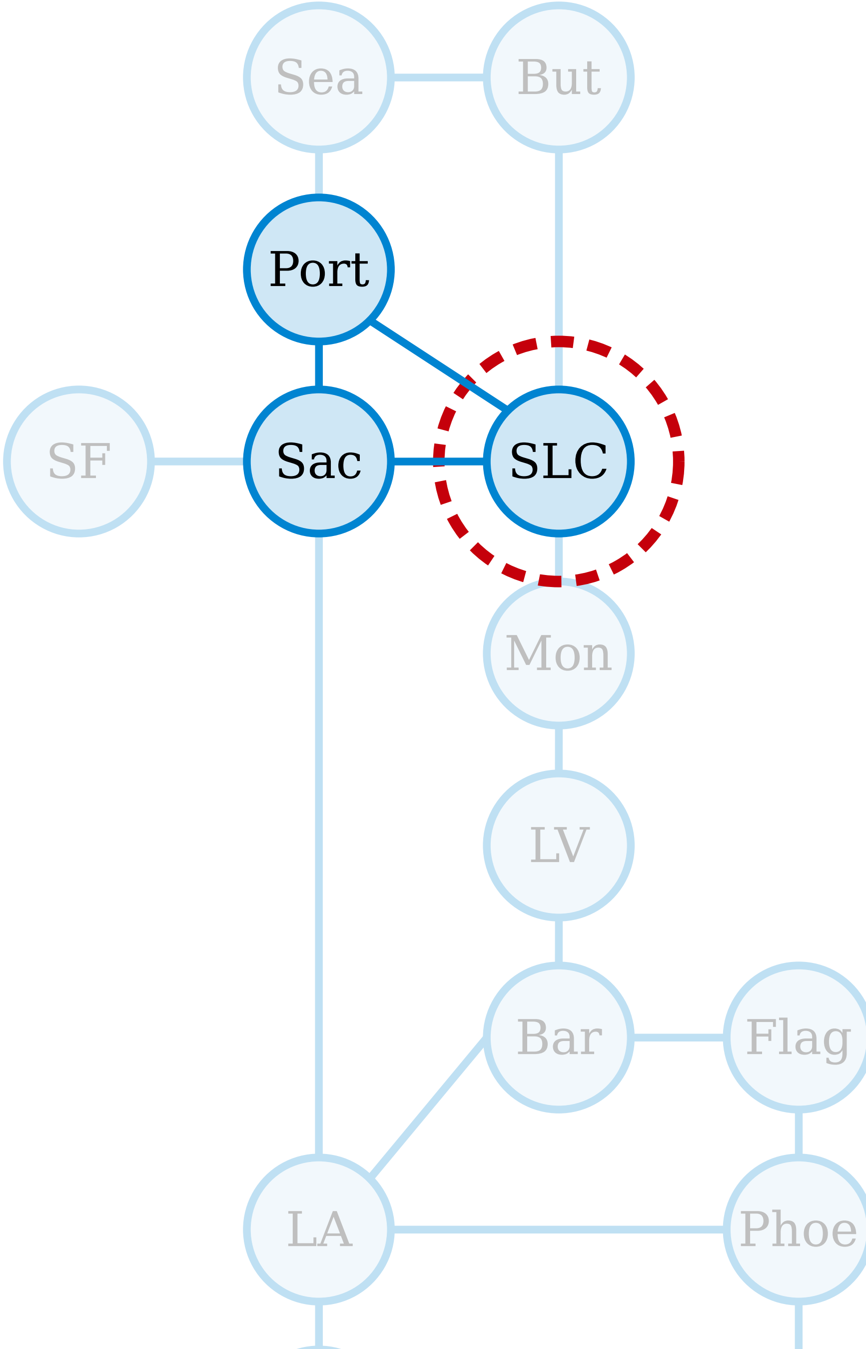


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.



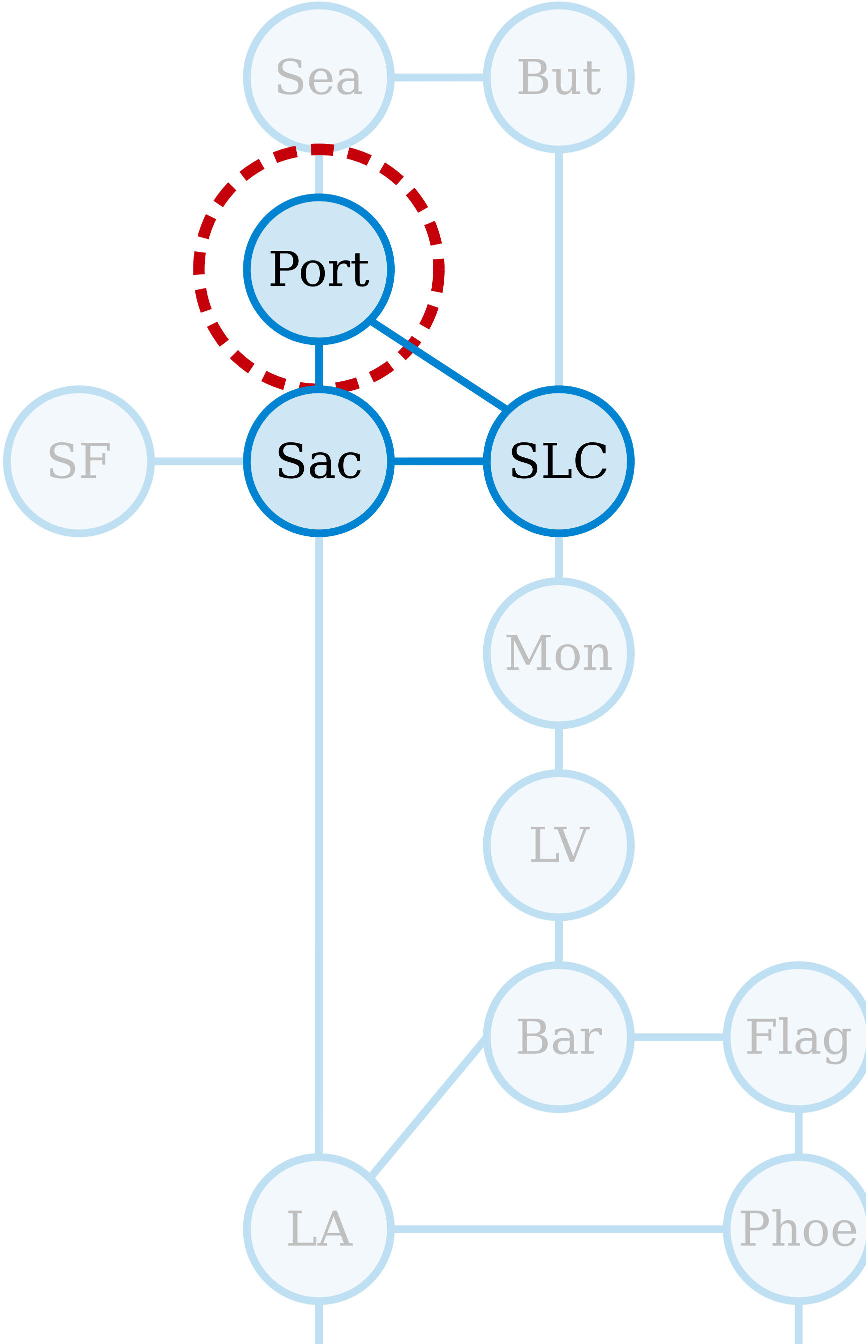
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.

Sac, SLC, Port, Sac, SLC



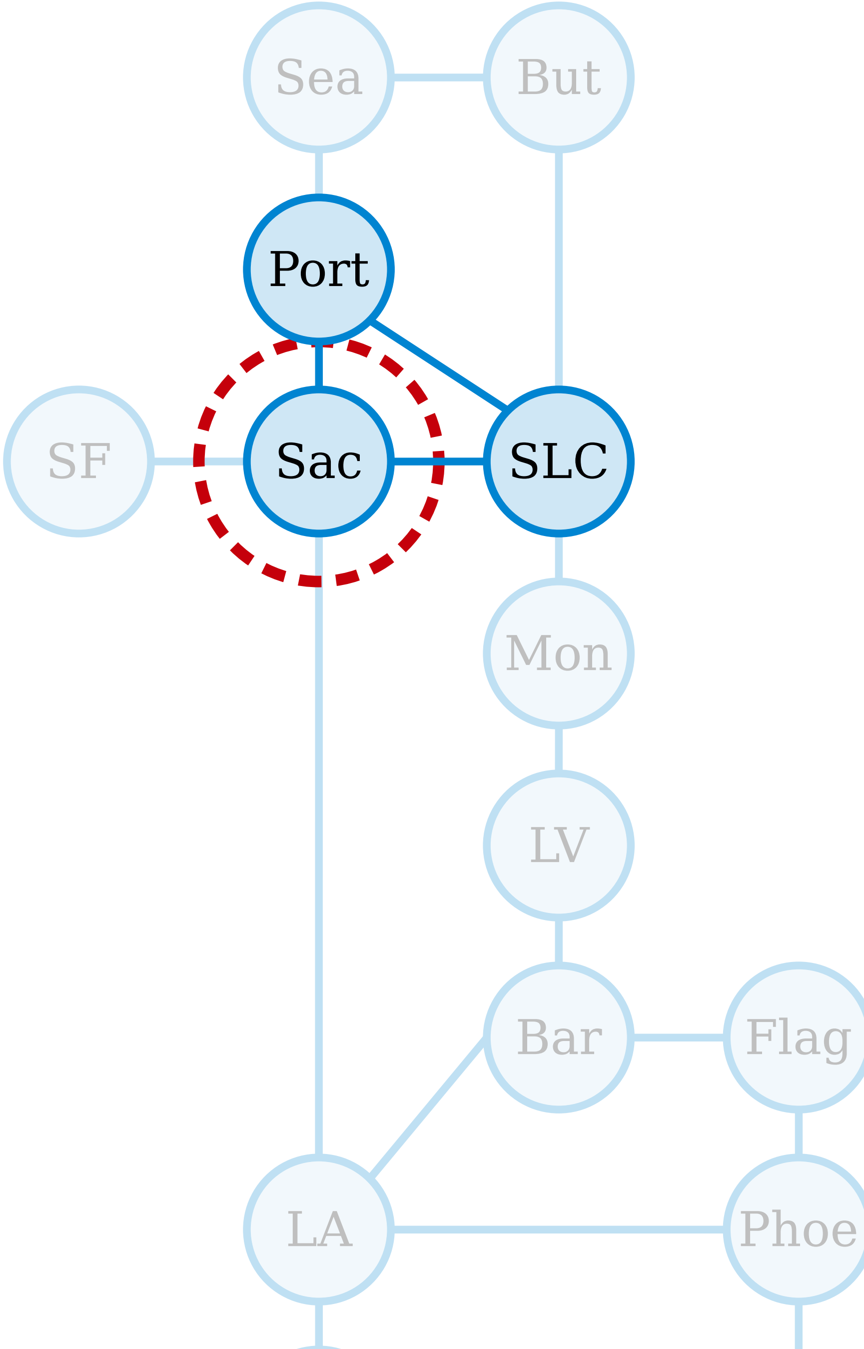
Sac, SLC, Port, Sac, SLC, Port

A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.



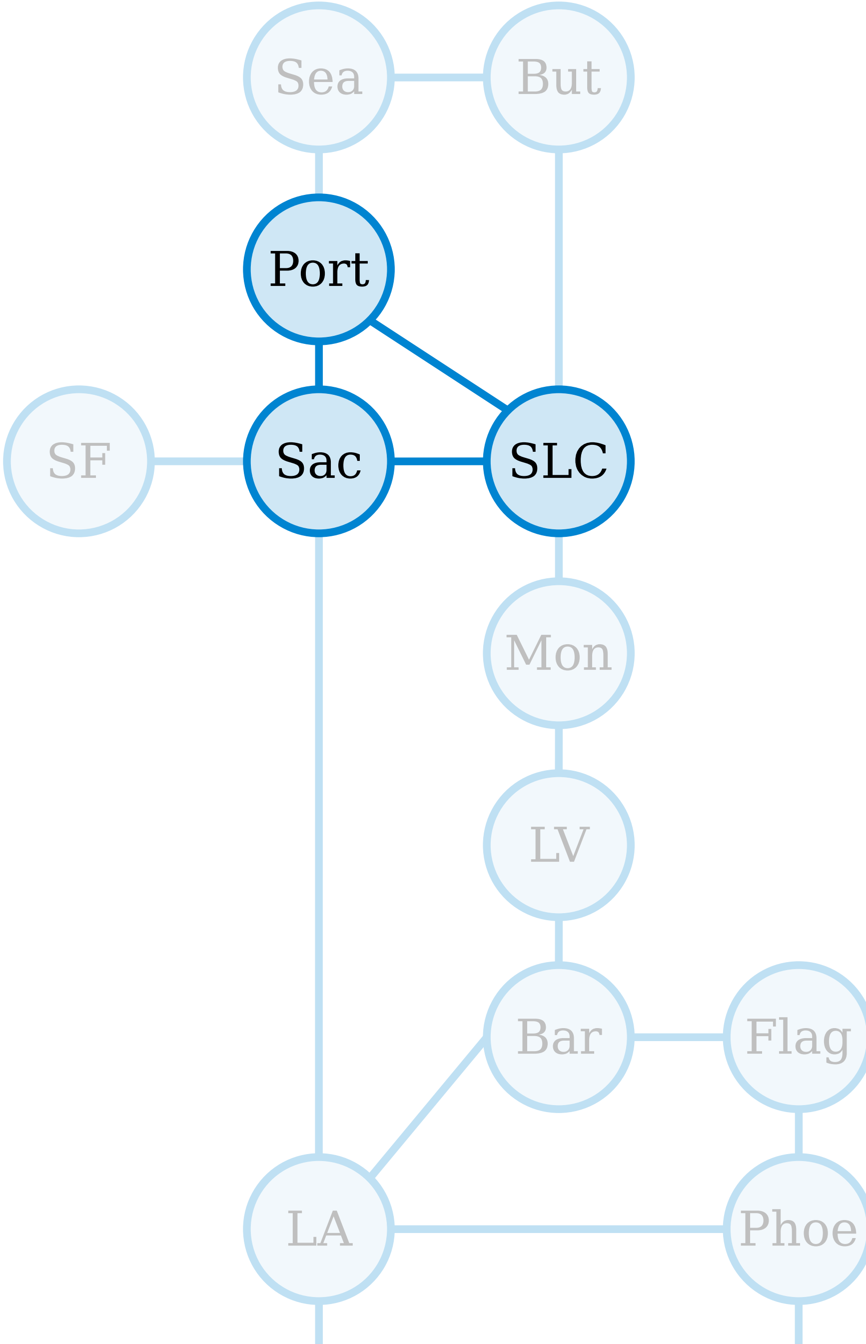
Sac, SLC, Port, Sac, SLC, Port, Sac

A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.



Sac, SLC, Port, Sac, SLC, Port, Sac

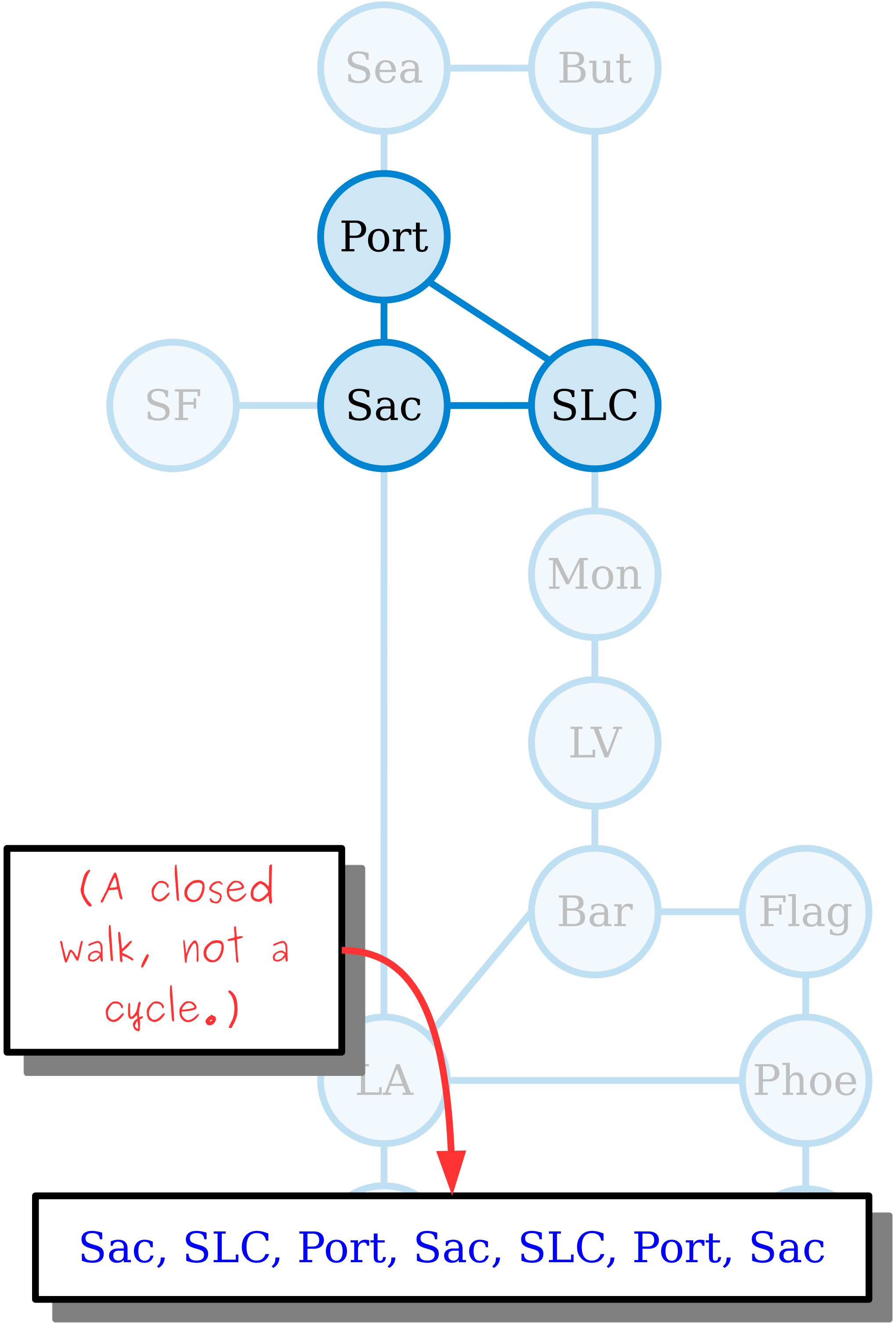
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.

A ***cycle*** in a graph is a closed walk that does not repeat any nodes or edges except the first/last node.



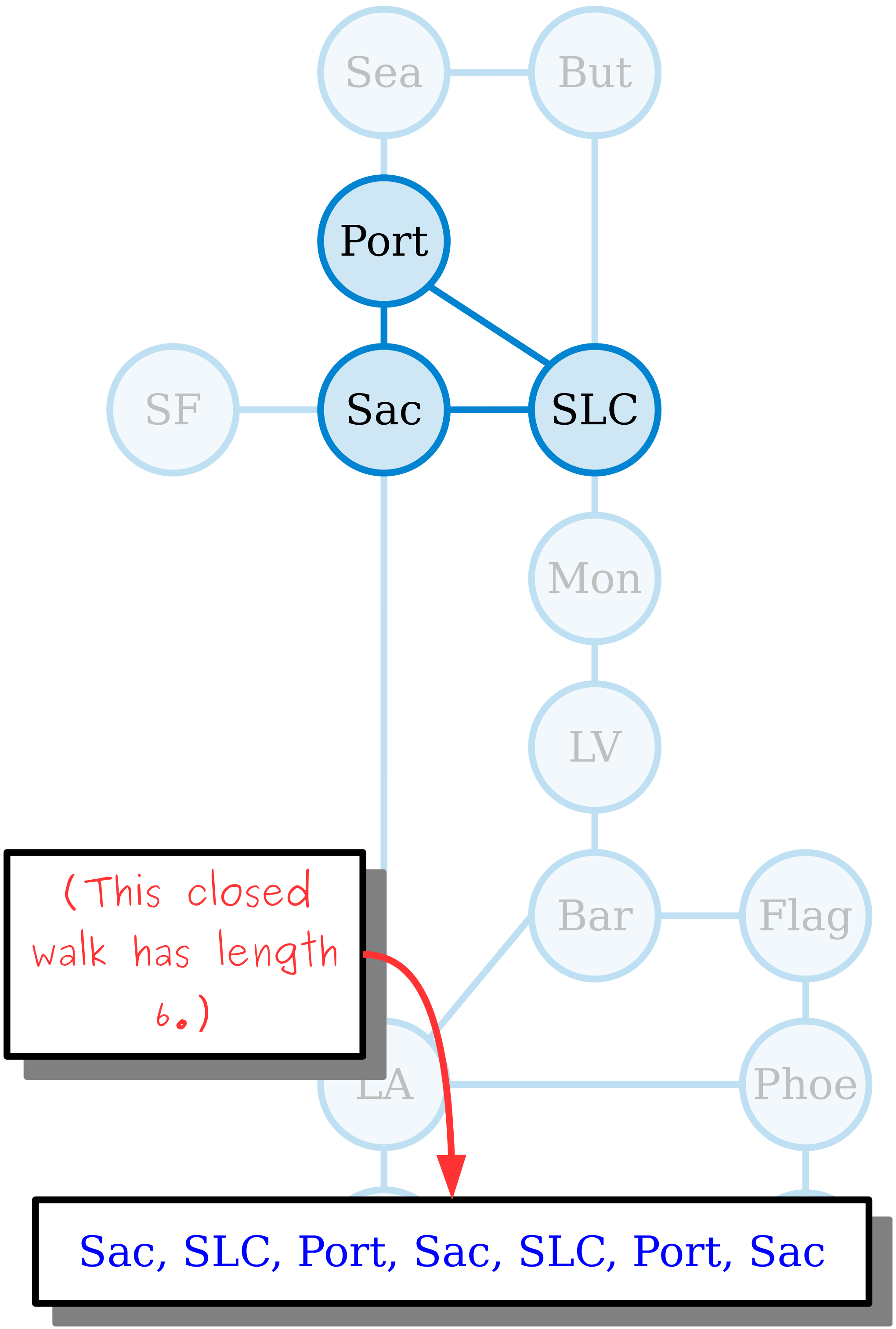
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.

A ***cycle*** in a graph is a closed walk that does not repeat any nodes or edges except the first/last node.



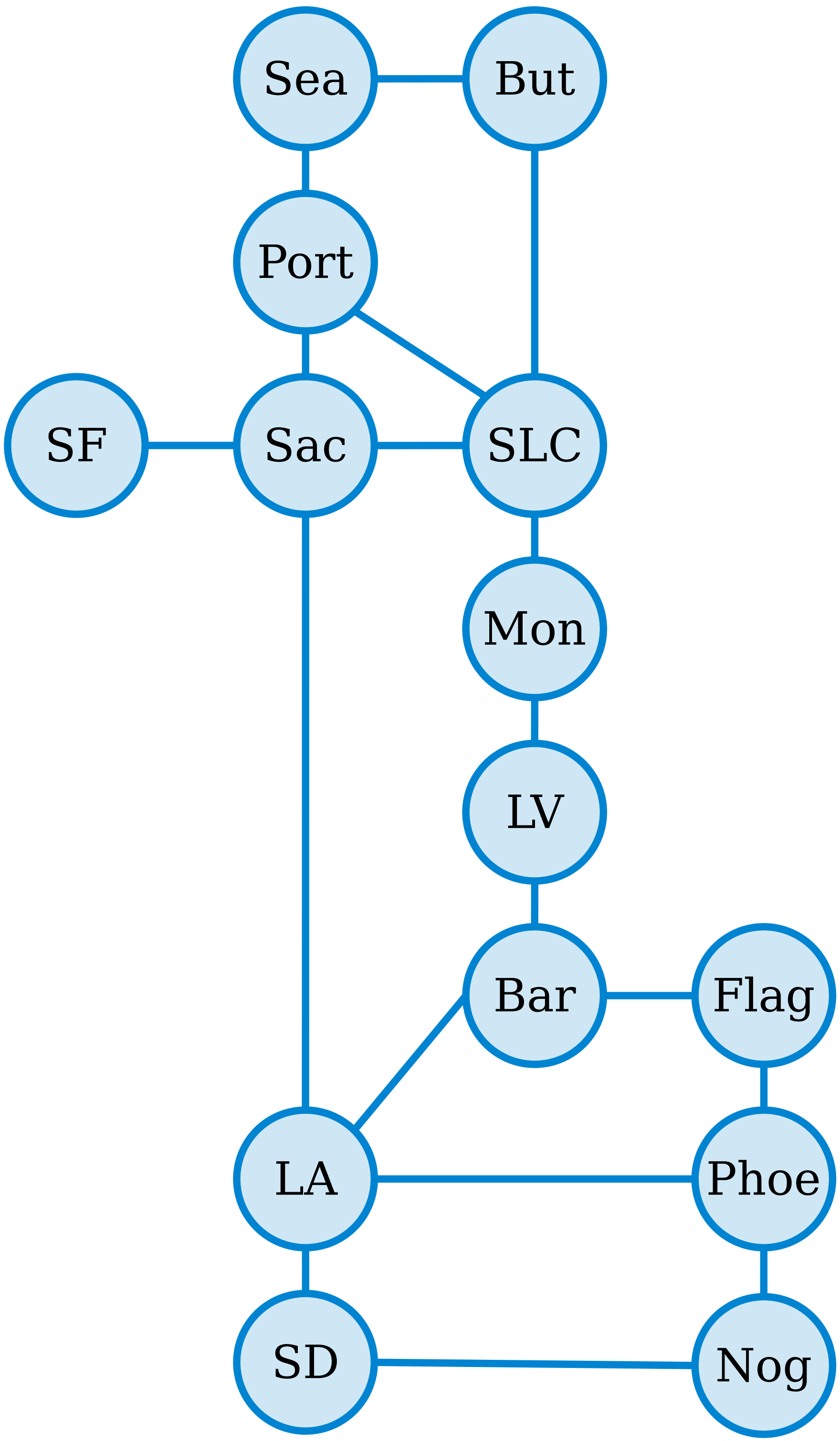
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.

A ***cycle*** in a graph is a closed walk that does not repeat any nodes or edges except the first/last node.



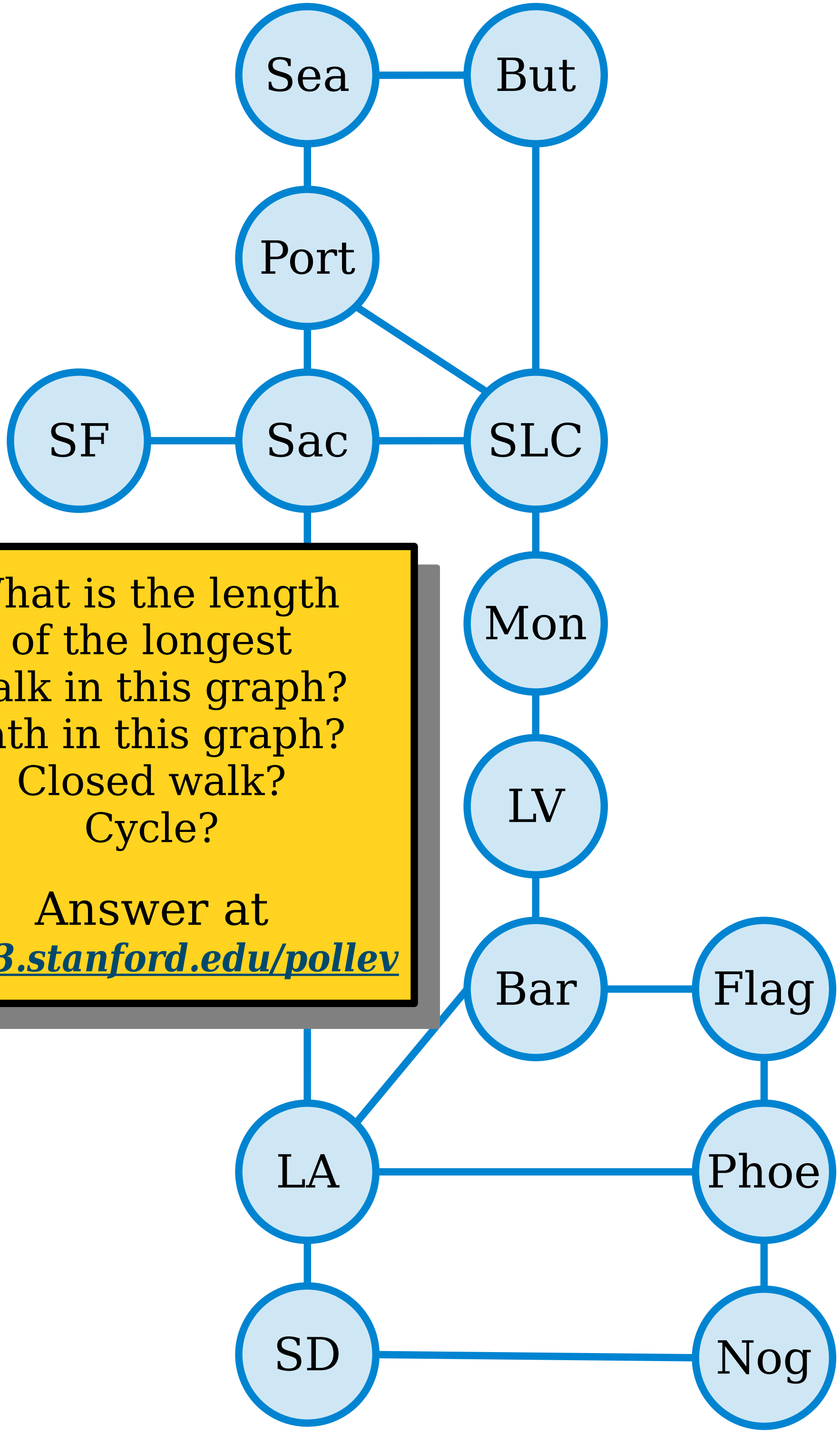
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

A ***path*** in a graph is walk that does not repeat any nodes.

A ***cycle*** in a graph is a closed walk that does not repeat any nodes or edges except the first/last node.



What is the length of the longest walk in this graph?
Path in this graph?
Closed walk?
Cycle?

Answer at
cs103.stanford.edu/pollev

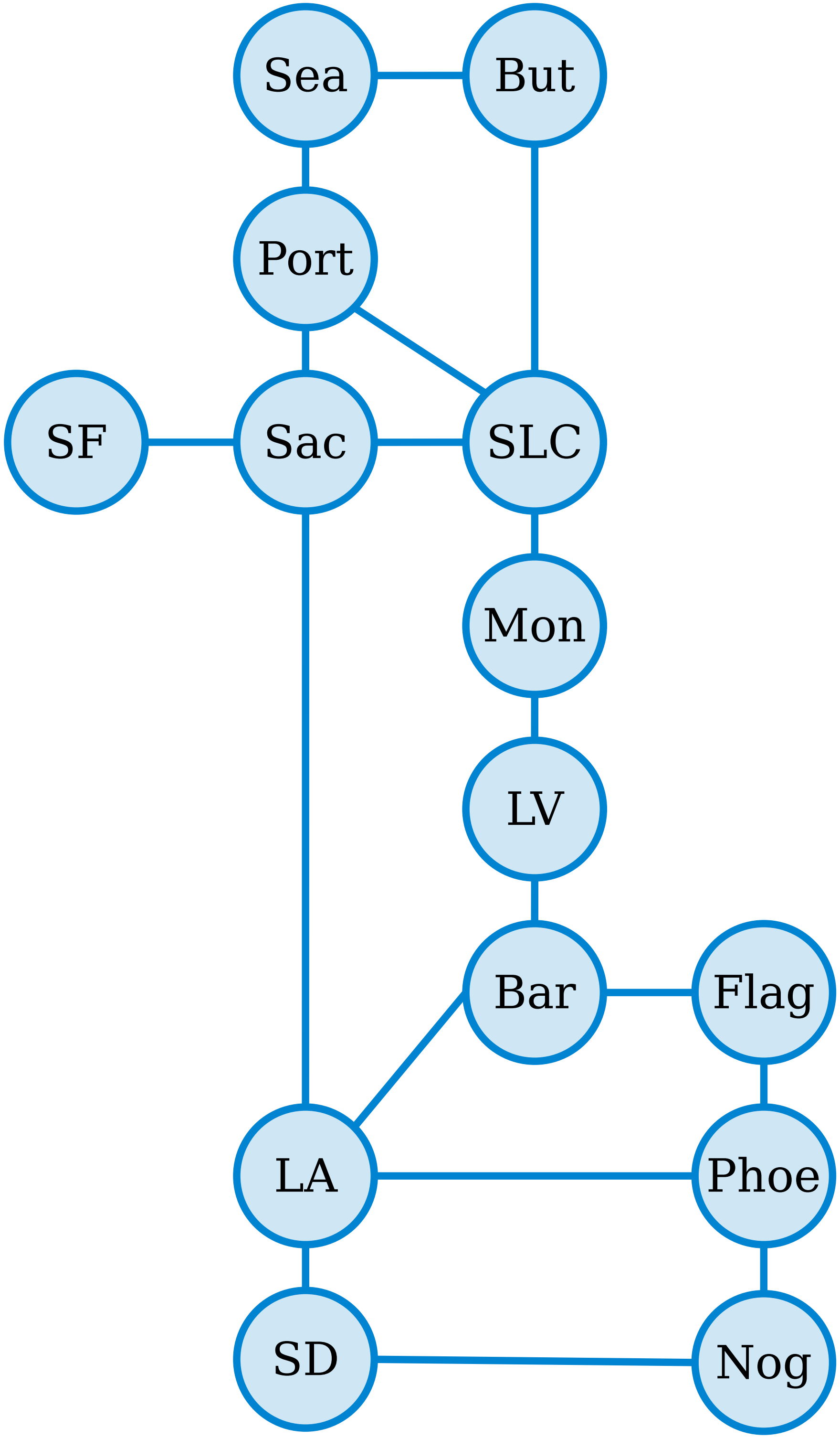
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the walk $v_1, \ldots, v_n$ is $n - 1$.

A ***closed walk*** in a graph is a walk from a node back to itself. (By convention, a closed walk cannot have length zero.)

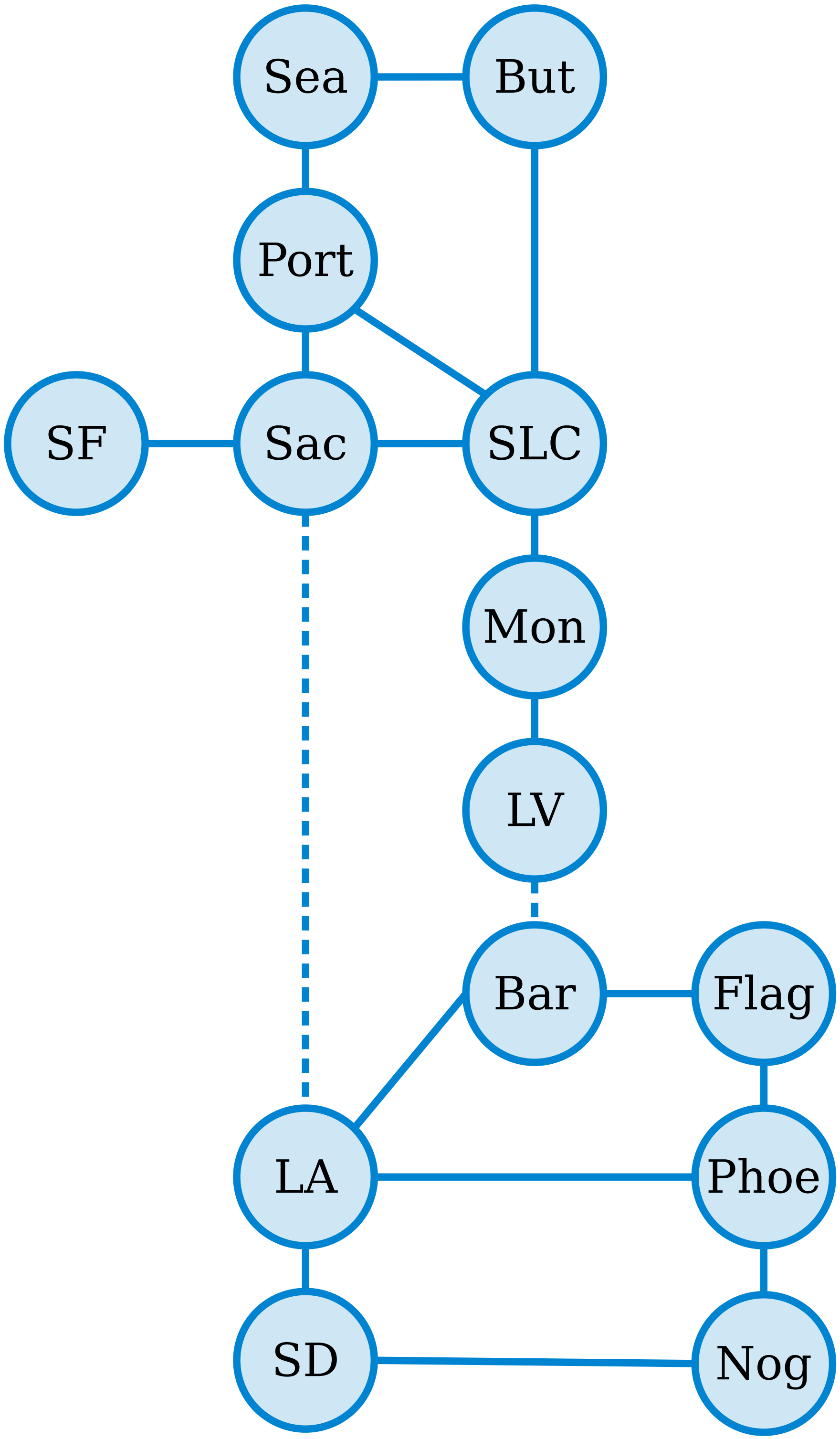
A ***path*** in a graph is walk that does not repeat any nodes.

A ***cycle*** in a graph is a closed walk that does not repeat any nodes or edges except the first/last node.



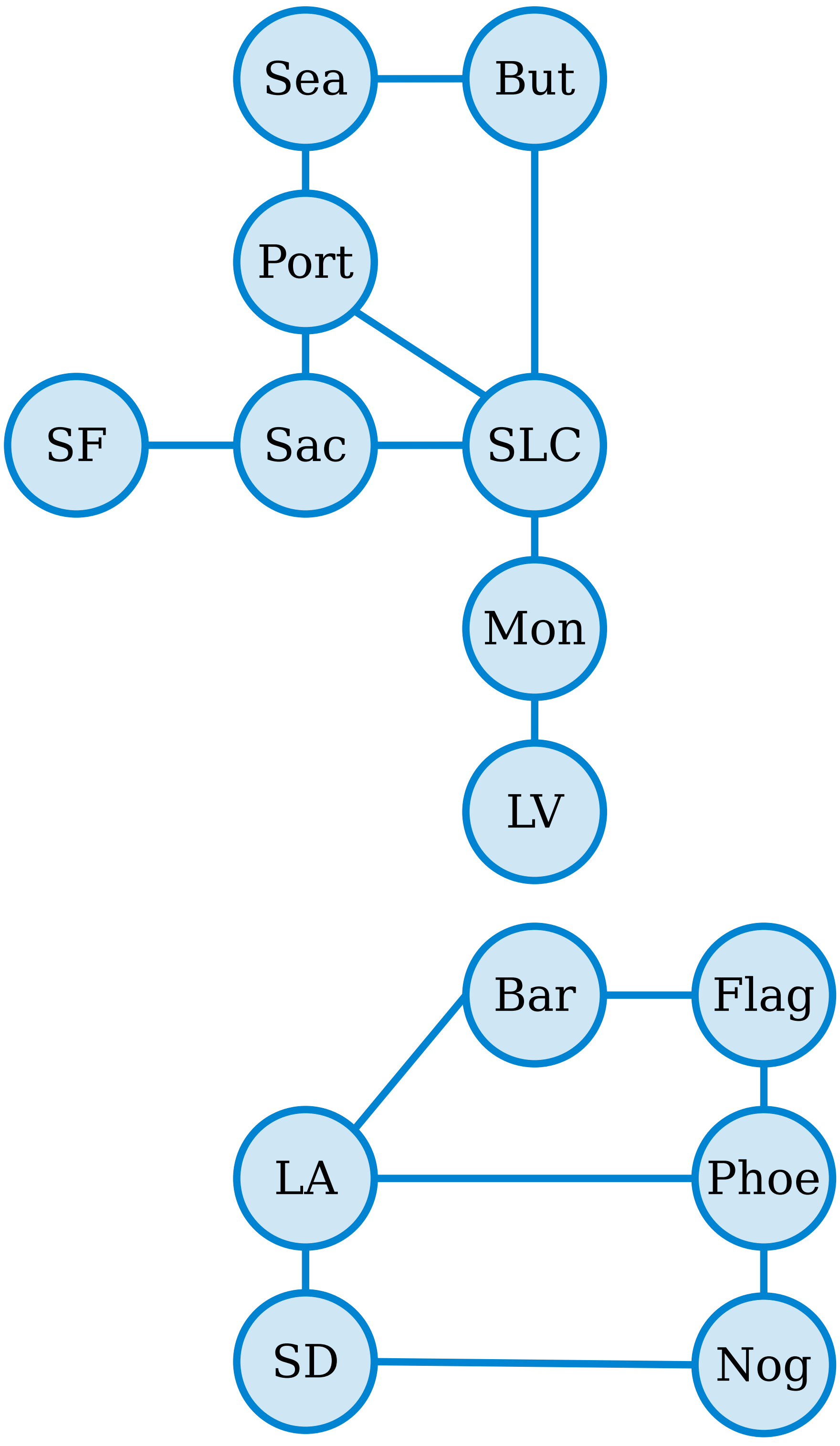
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

A ***path*** in a graph is walk that does not repeat any nodes.



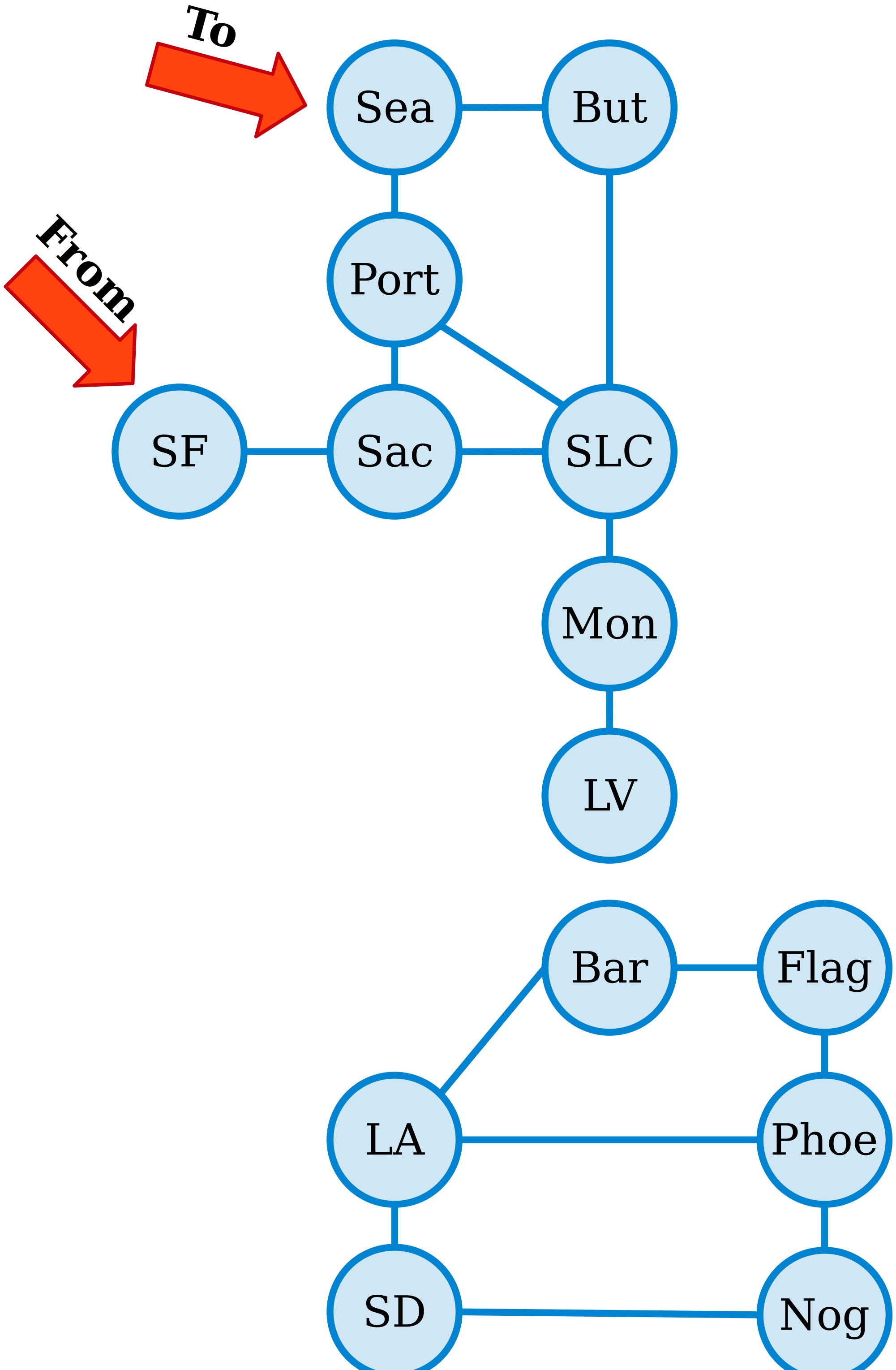
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

A ***path*** in a graph is walk that does not repeat any nodes.



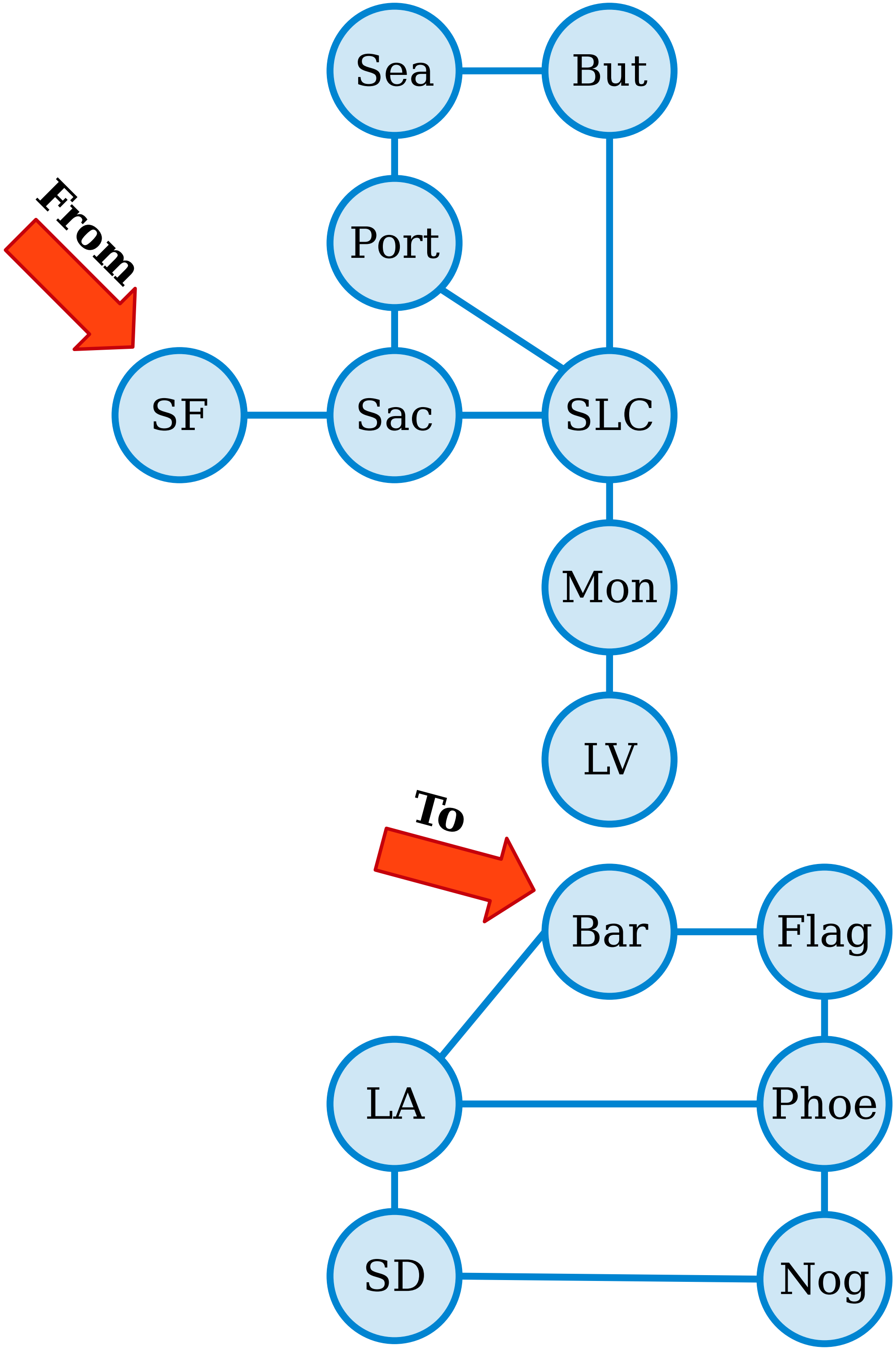
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

A ***path*** in a graph is walk that does not repeat any nodes.



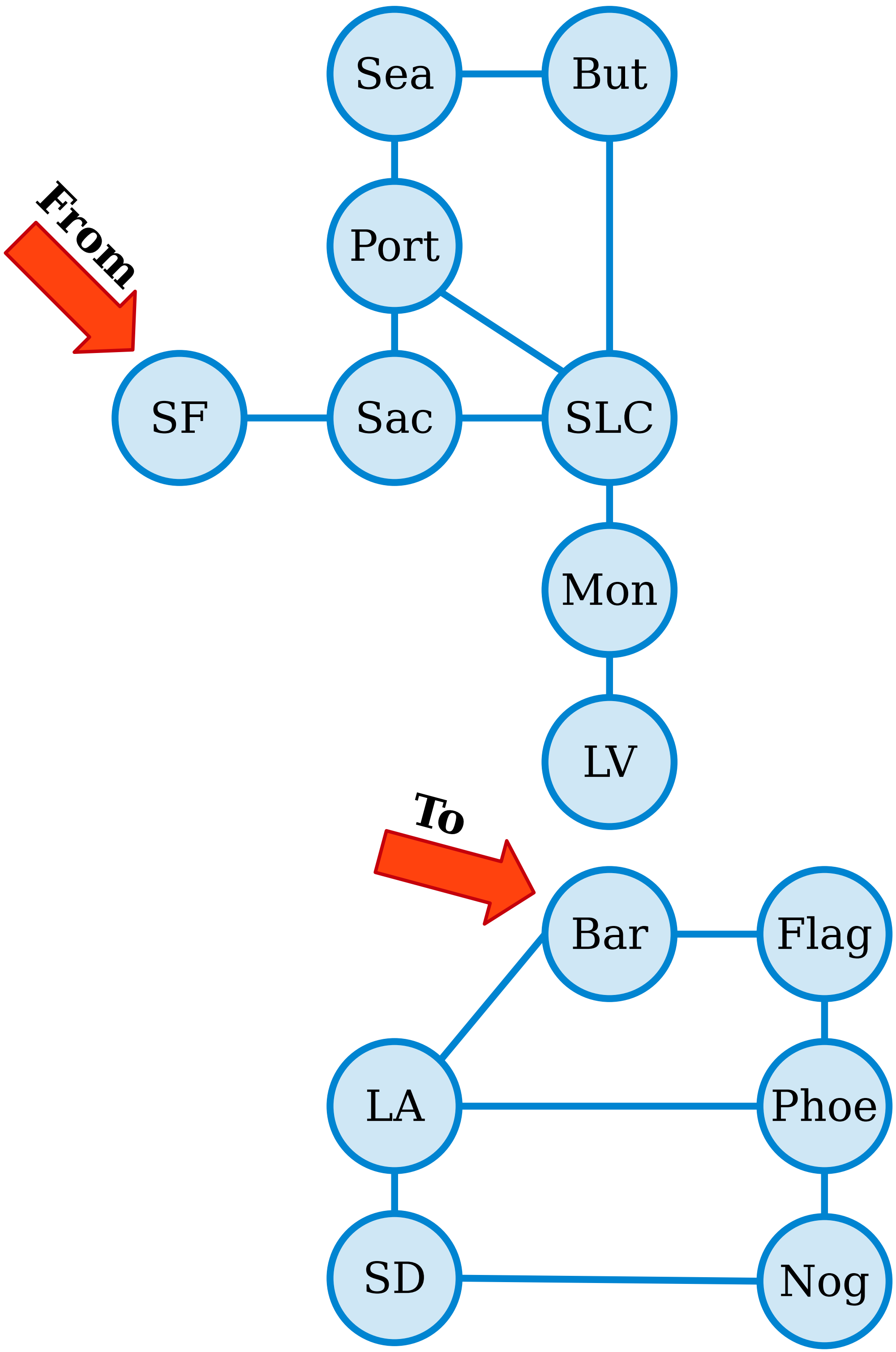
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

A ***path*** in a graph is walk that does not repeat any nodes.



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

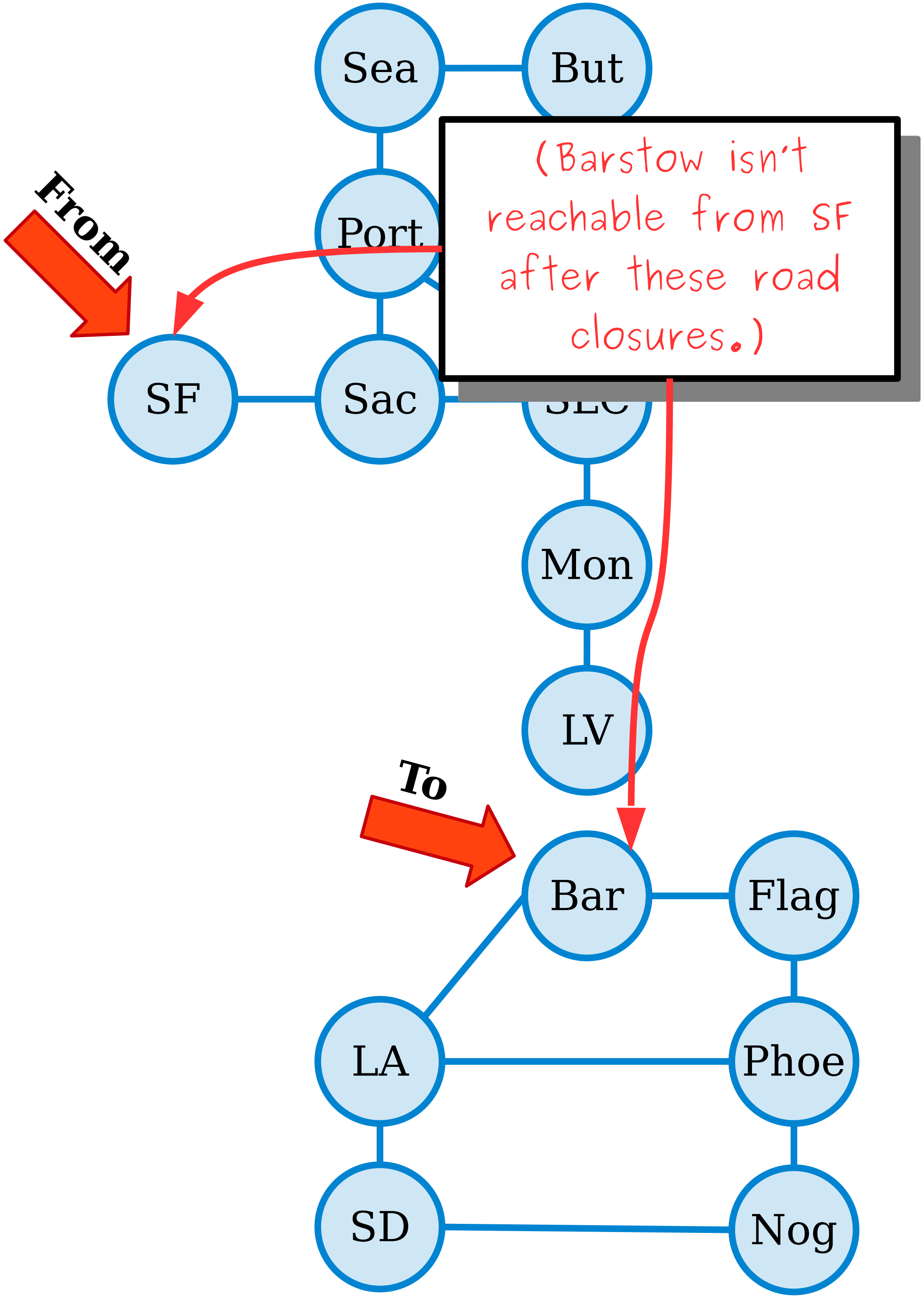
A ***path*** in a graph is walk that does not repeat any nodes.



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

A ***path*** in a graph is walk that does not repeat any nodes.

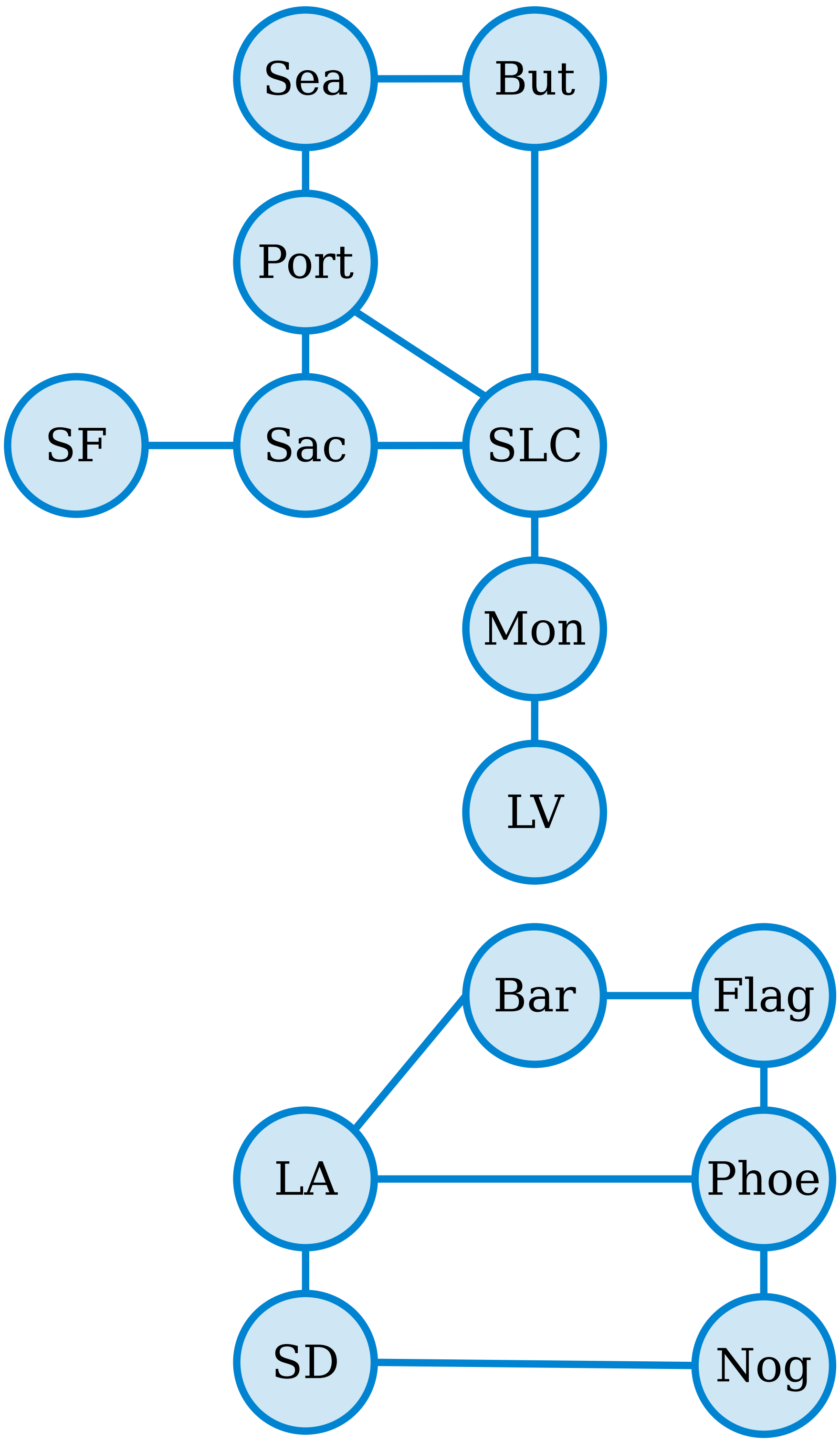
A node v is ***reachable*** from a node u if there is a path from u to v.



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

A ***path*** in a graph is walk that does not repeat any nodes.

A node v is ***reachable*** from a node u if there is a path from u to v.

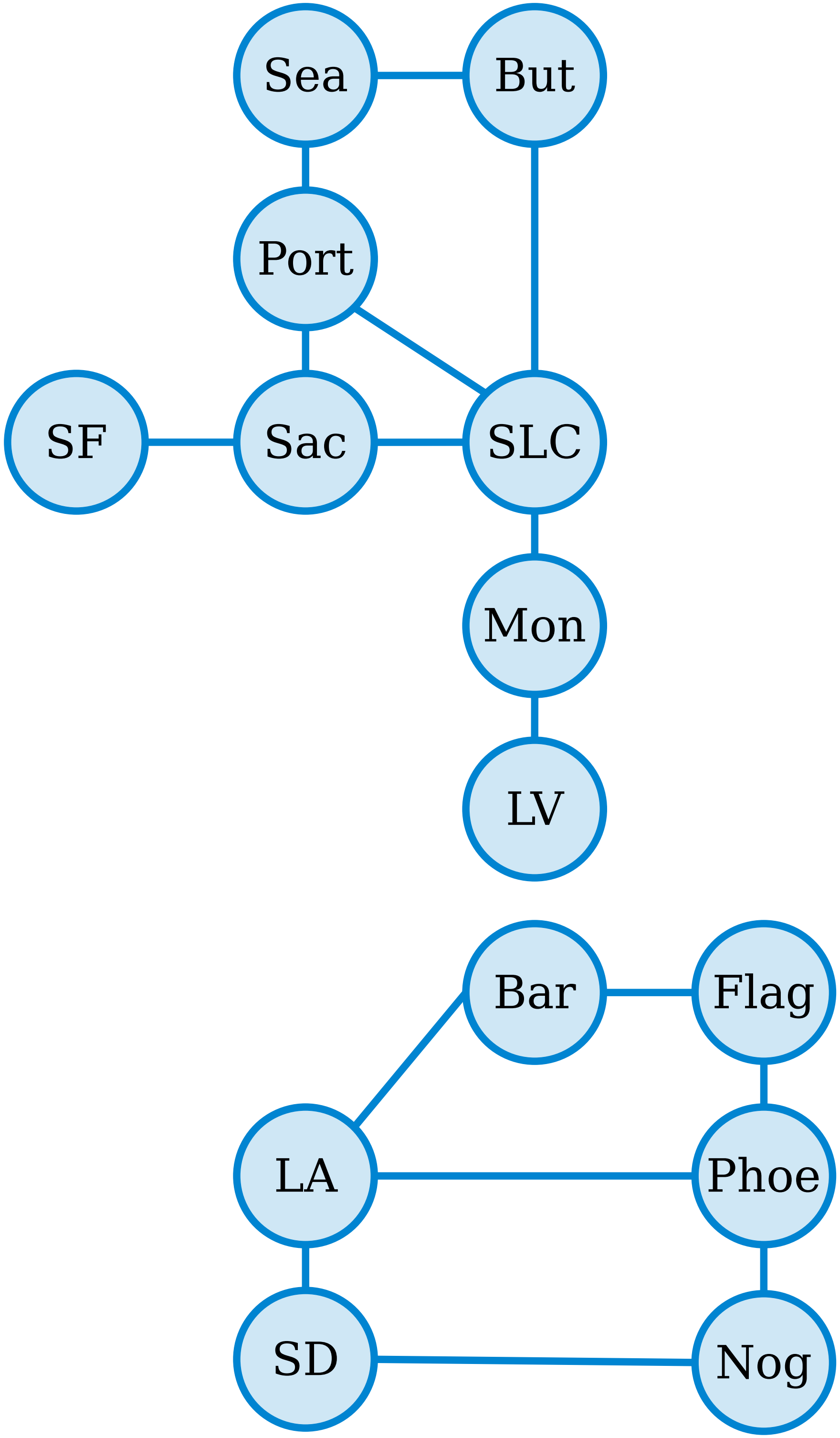


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

A ***path*** in a graph is walk that does not repeat any nodes.

A node v is ***reachable*** from a node u if there is a path from u to v.

A graph G is called ***connected*** if all pairs of distinct nodes in G are reachable.



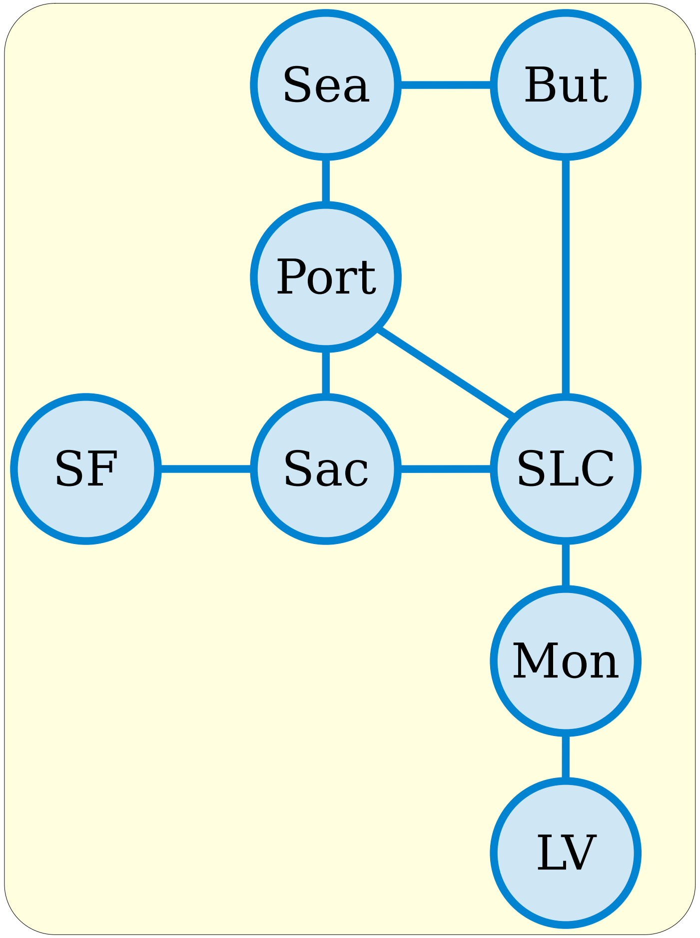
A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

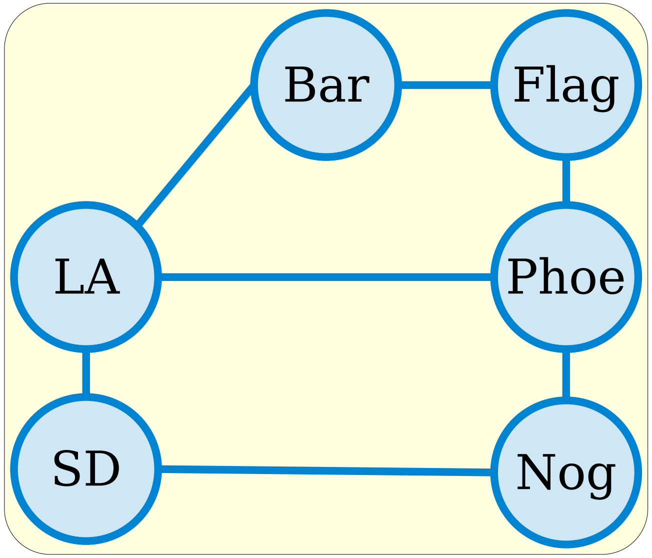
A ***path*** in a graph is walk that does not repeat any nodes.

A node v is ***reachable*** from a node u if there is a path from u to v.

A graph G is called ***connected*** if all pairs of distinct nodes in G are reachable.

(This graph is not connected.)



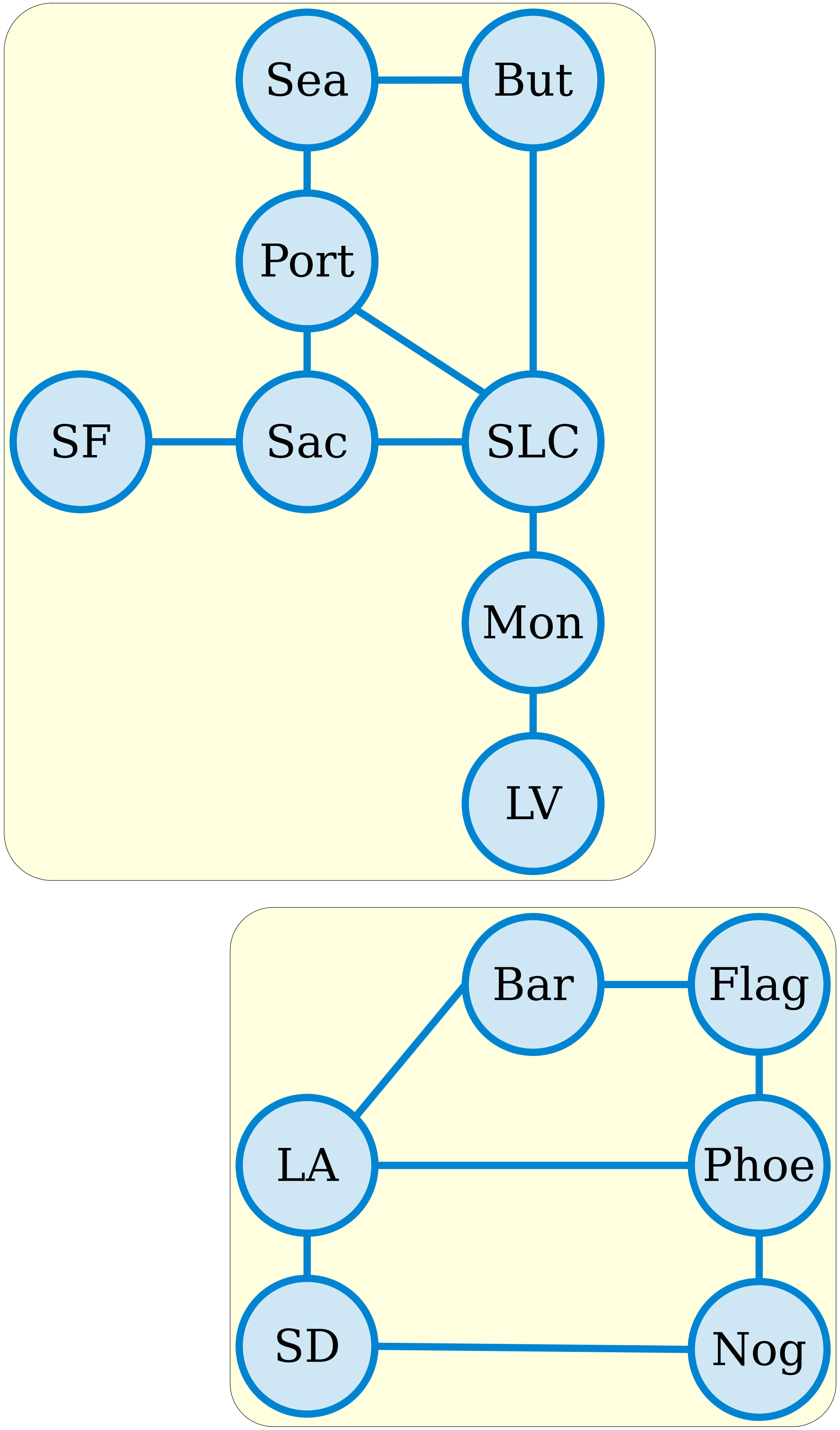


A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

A ***path*** in a graph is walk that does not repeat any nodes.

A node v is ***reachable*** from a node u if there is a path from u to v.

A graph G is called ***connected*** if all pairs of distinct nodes in G are reachable.



A ***walk*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

A ***path*** in a graph is walk that does not repeat any nodes.

A node v is ***reachable*** from a node u if there is a path from u to v.

A graph G is called ***connected*** if all pairs of distinct nodes in G are reachable.

A ***connected component*** (or ***CC***) of G is a set consisting of a node and every node reachable from it.

Fun Facts

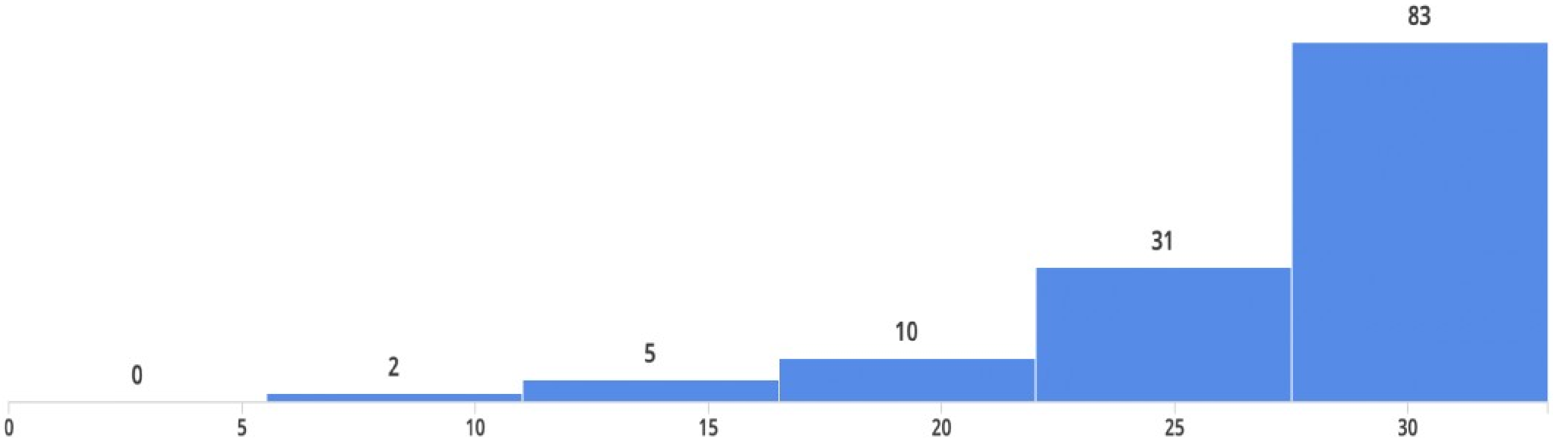
- Here's a collection of useful facts about graphs that you can take as a given.
  - ***Theorem:*** If $G = (V, E)$ is a graph and $u, v \in V$, then there is a path from u to v if and only if there's a walk from u to v.
  - ***Theorem:*** If G is a graph and C is a cycle in G, then C's length is at least three and C contains at least three nodes.
  - ***Theorem:*** If $G = (V, E)$ is a graph, then every node in V belongs to exactly one connected component of G.
  - ***Theorem:*** If $G = (V, E)$ is a graph, then G is not connected if and only if G has two or more connected components.
- Looking for more practice working with formal definitions? Prove these results!

Fun Facts

- Here's a collection of useful facts about graphs that you can take as a given.
  - ***Theorem:*** If $G = (V, E)$ is a graph and $u, v \in V$, then there is a path from u to v if and only if there's a walk from u to v.
  - ***Theorem:*** If G is a graph and C is a cycle in G, then C's length is at least three and C contains at least three nodes.
  - ***Theorem:*** If $G = (V, E)$ is a graph, then every node in V belongs to exactly one connected component of G.
  - ***Theorem:*** If $G = (V, E)$ is a graph, then G is not connected if and only if G has two or more connected components.
- Looking for more practice working with formal definitions? Prove these results!

Time-Out for Announcements!

Problem Set Two Graded



Distribution (written part)
Median: 88%

Midterm Exam Logistics

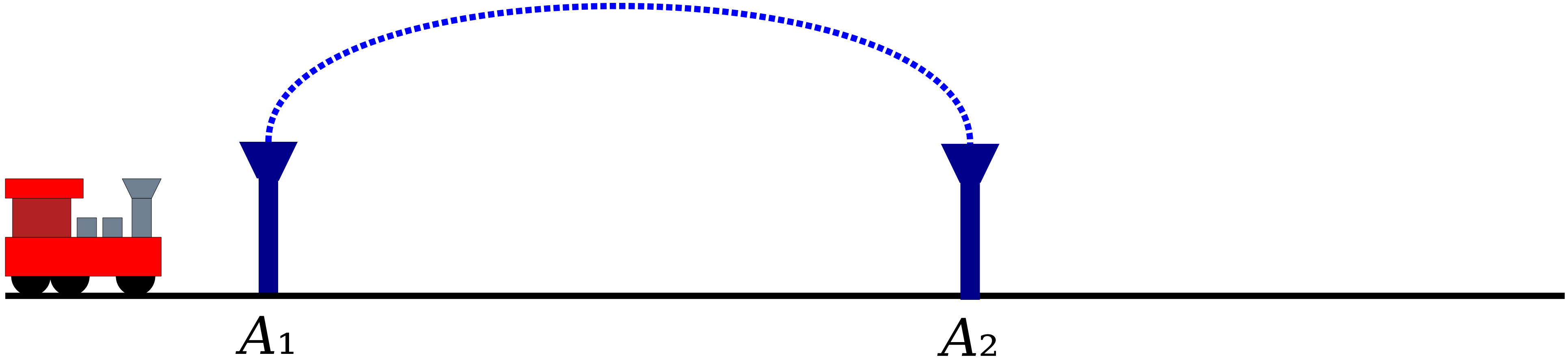
- Our first midterm exam is next Tuesday, April 30th from 7:00PM – 10:00PM.
  - Check the course website for logistics.
- We will have a problem set on graph theory next week, but it's shorter than our usual problem sets because we know you have the midterm.
- We have reached out to everyone who will be taking the exam at an alternate time. If you intend to take the exam outside the normal time and haven't heard from us, contact us immediately.

Preparing for the Exam

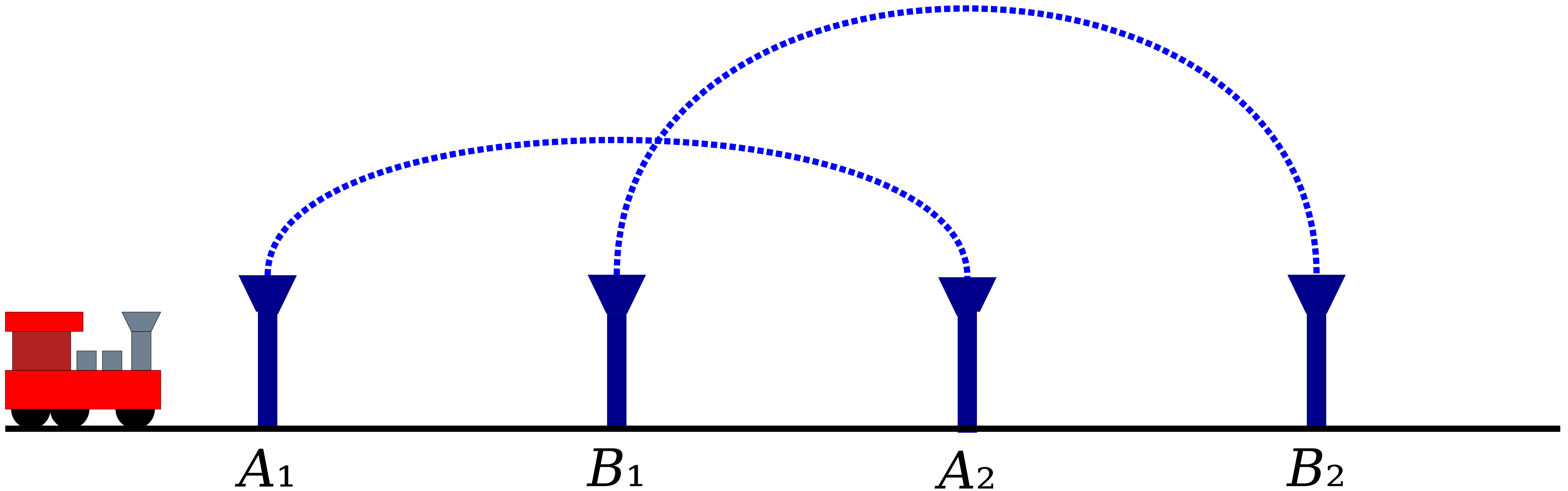
- Make sure to ***review your feedback*** on PS1 and PS2.
  - “Make new mistakes.”
  - Come talk to us if you have questions!
- There’s a huge bank of practice problems up on the course website.
- Best of luck – ***you can do this!***

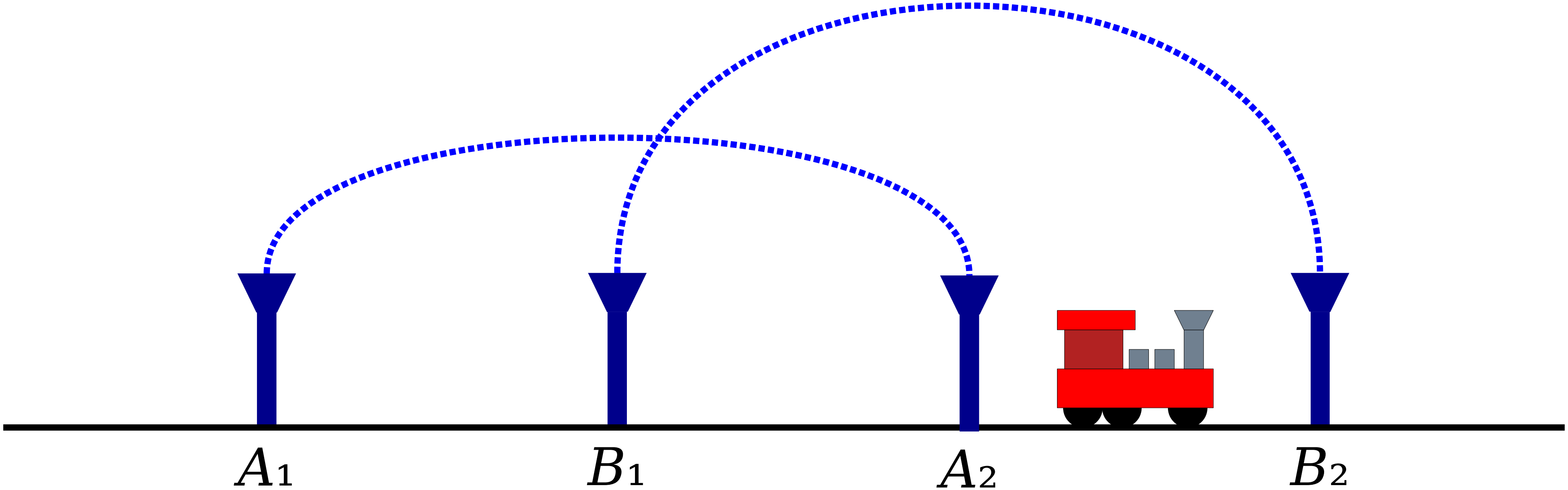
Back to CS103!

The Teleported Train Problem

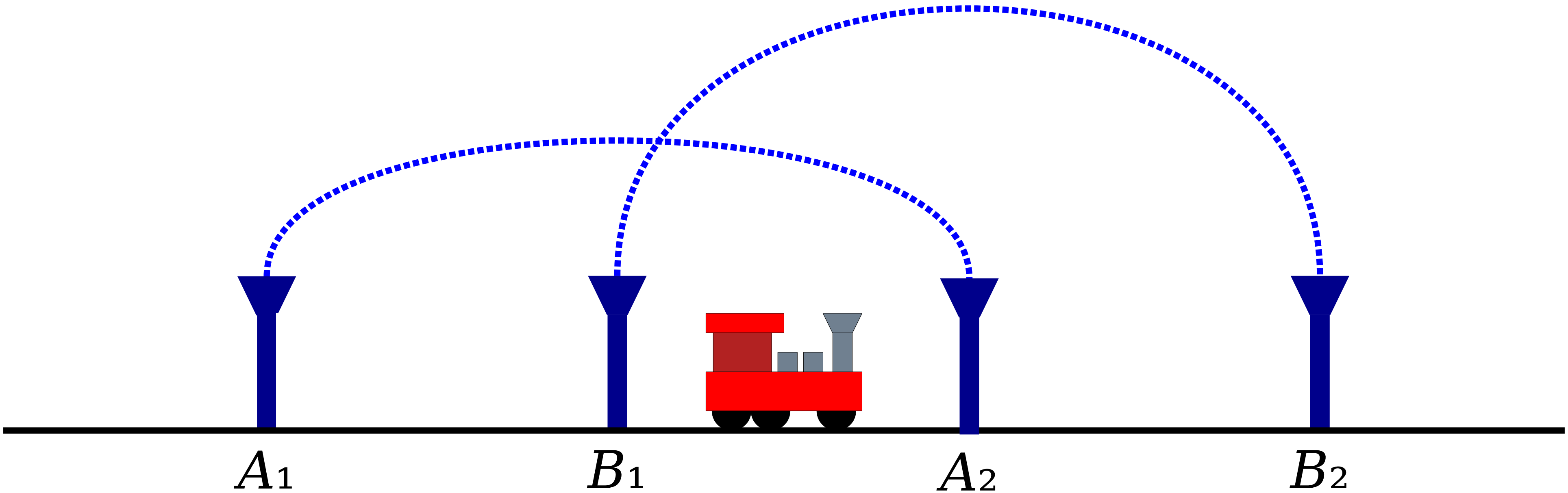


These are ***teleporters***.
Anything entering a teleporter from the left side emerges from the right side of the paired teleporter.

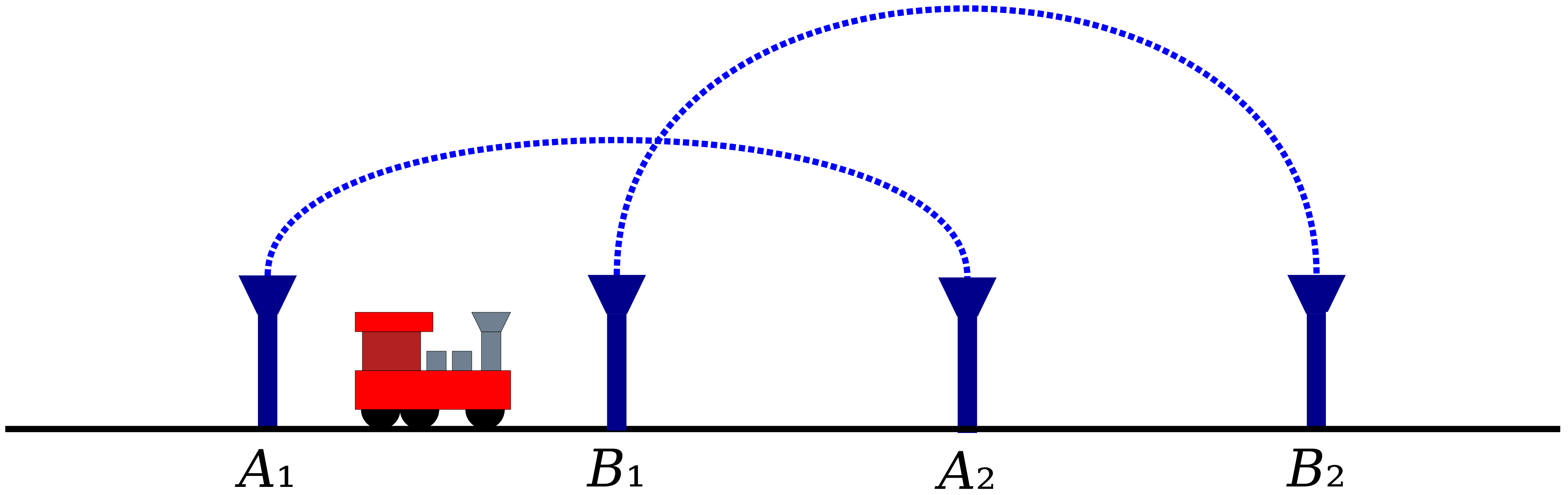

A_1
B_1
A_2
B_2



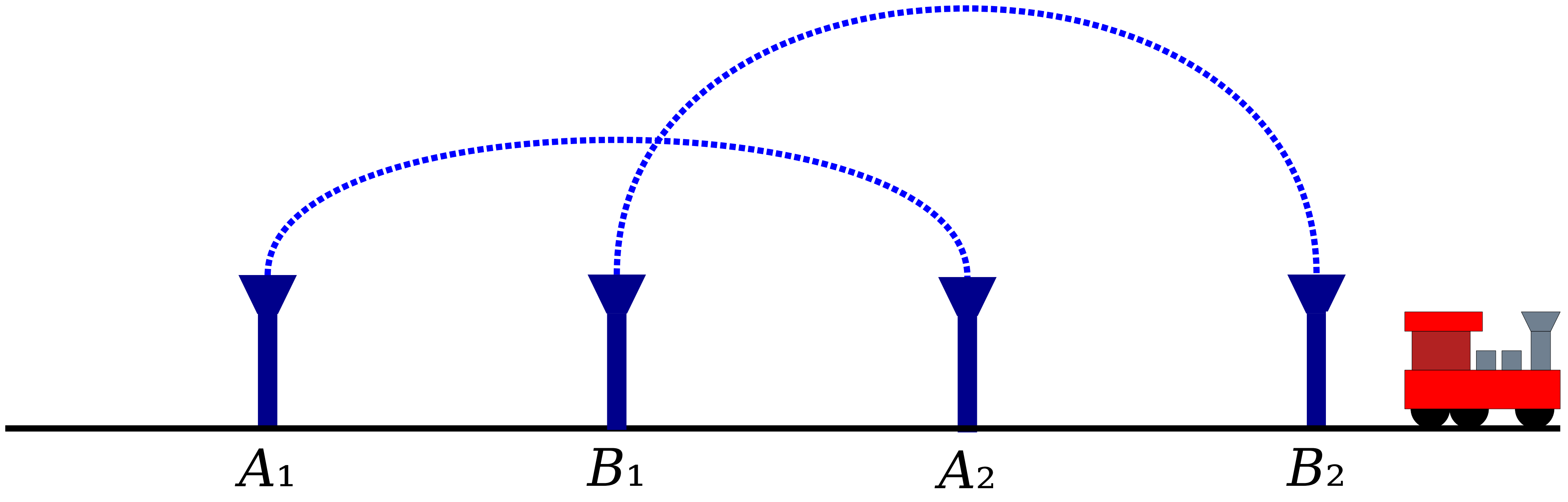
A_1
B_1
A_2
B_2



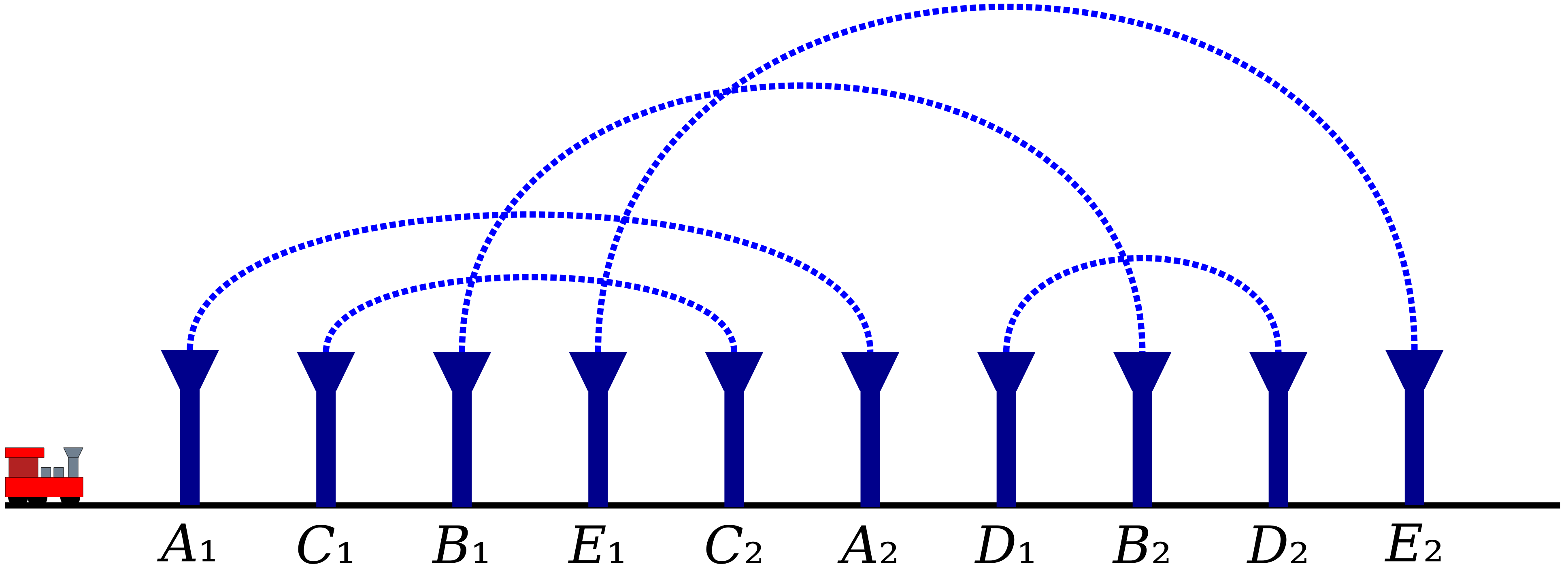
A_1
B_1
A_2
B_2



A_1
B_1
A_2
B_2

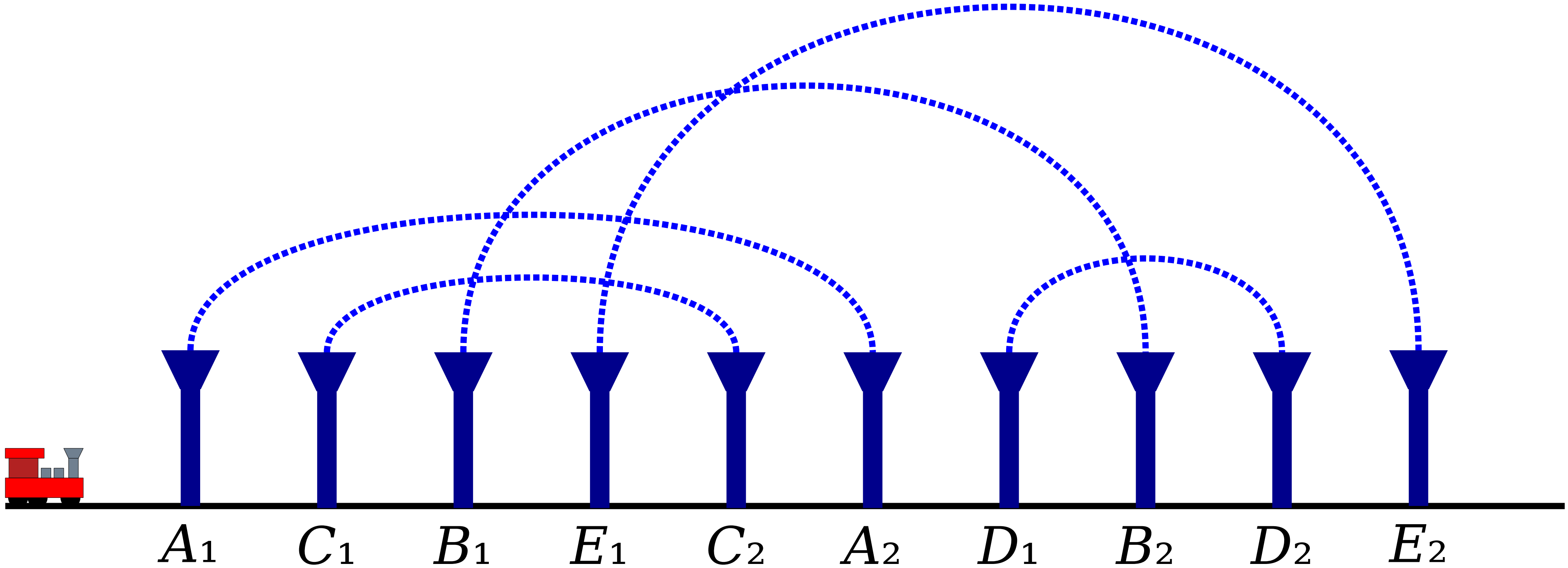


It took a while, but eventually the train reached the end of the track.

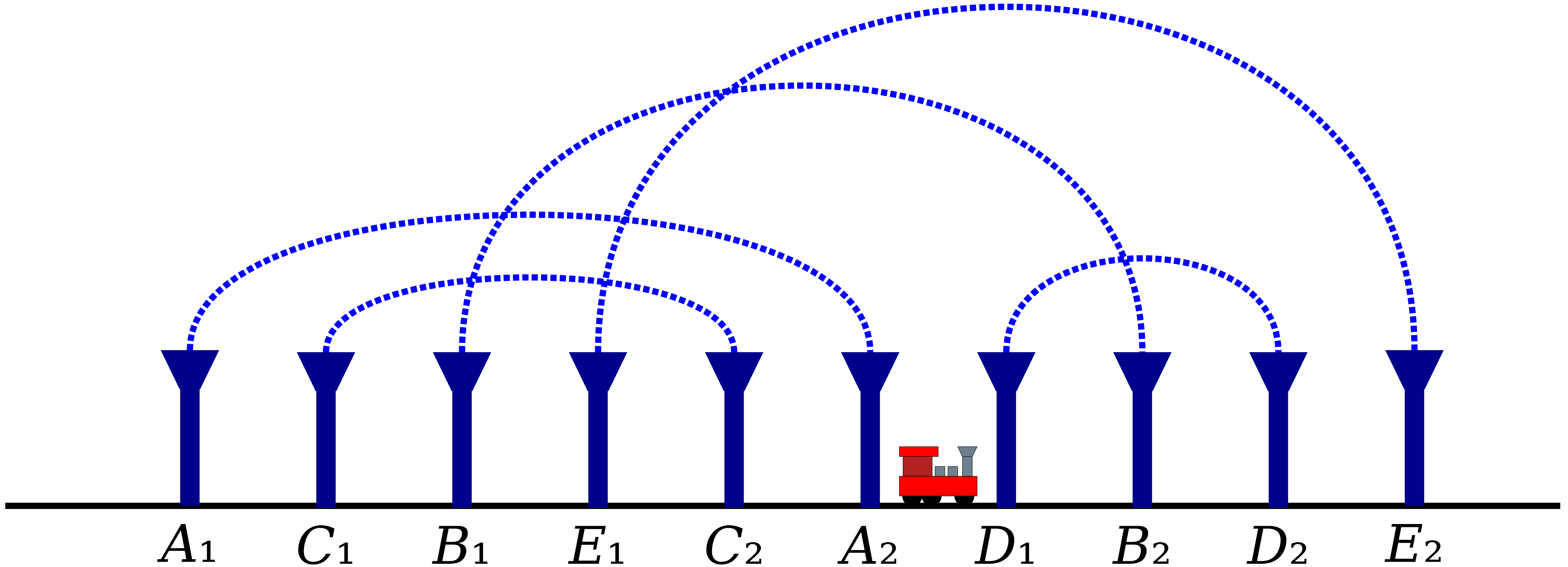


Will the train reach the end of the track? Or will it get stuck in a loop?

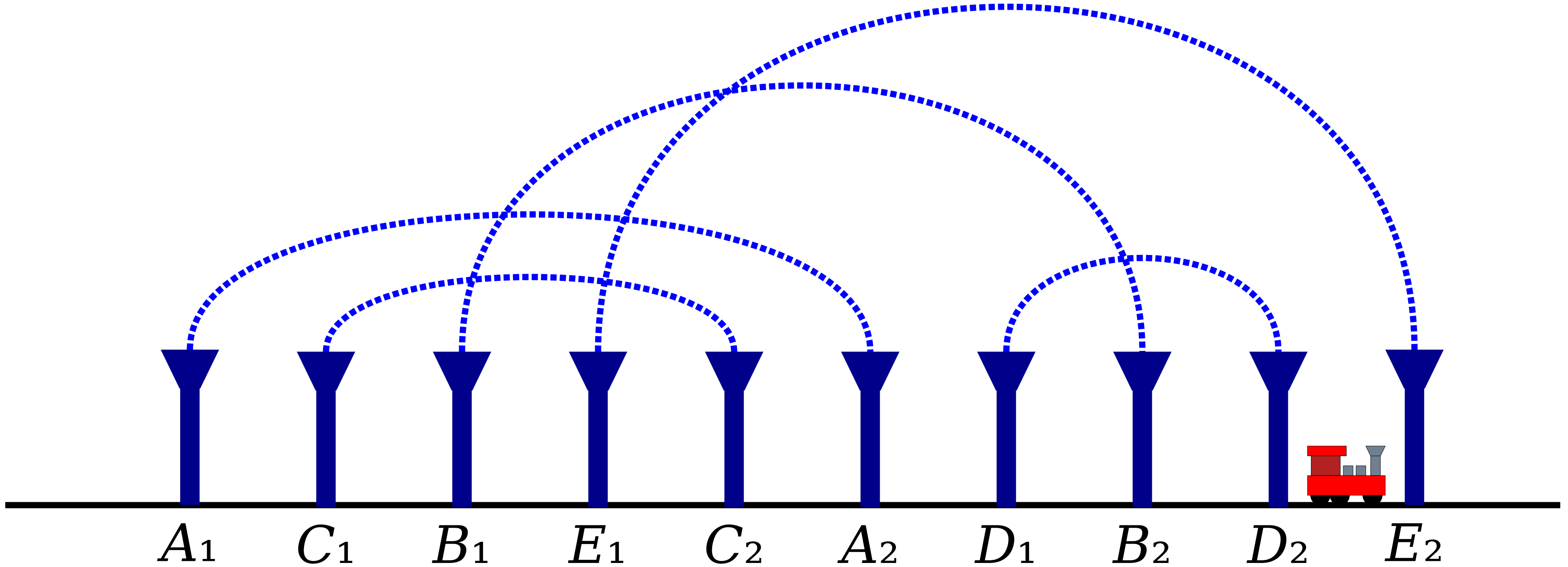
Answer at
https://cs103.stanford.edu/pollev



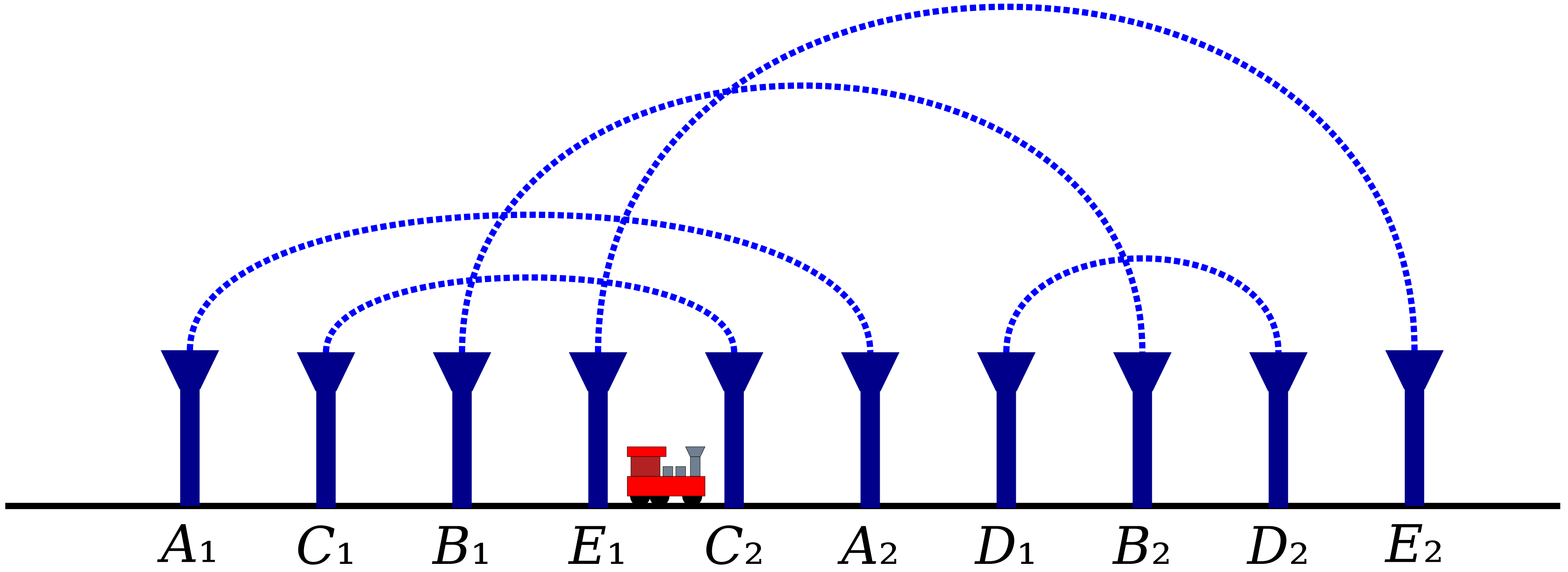
A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2



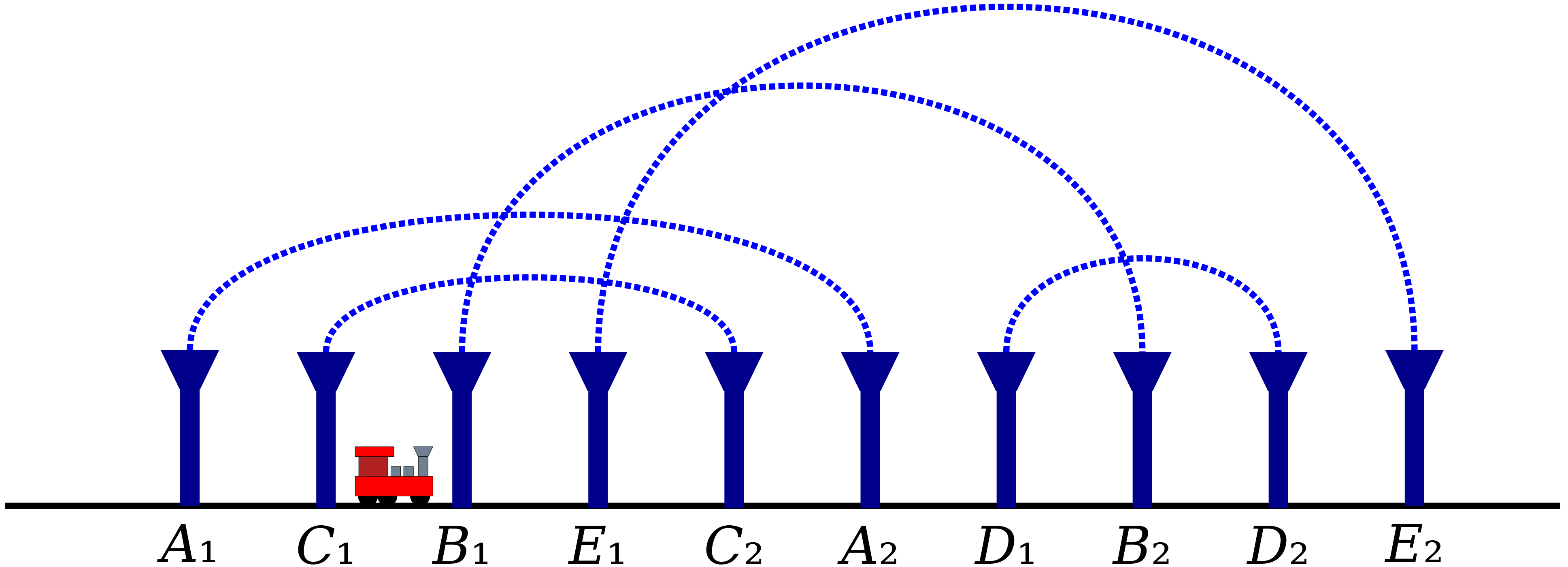
A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2



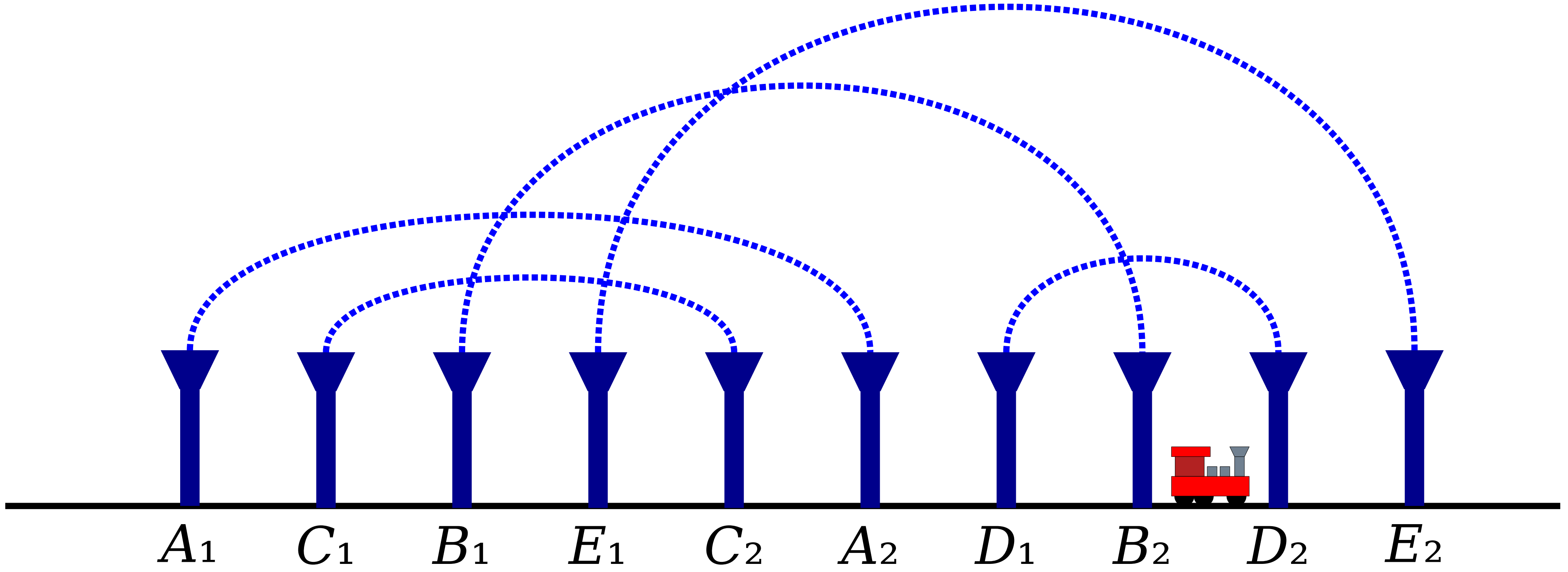
A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2



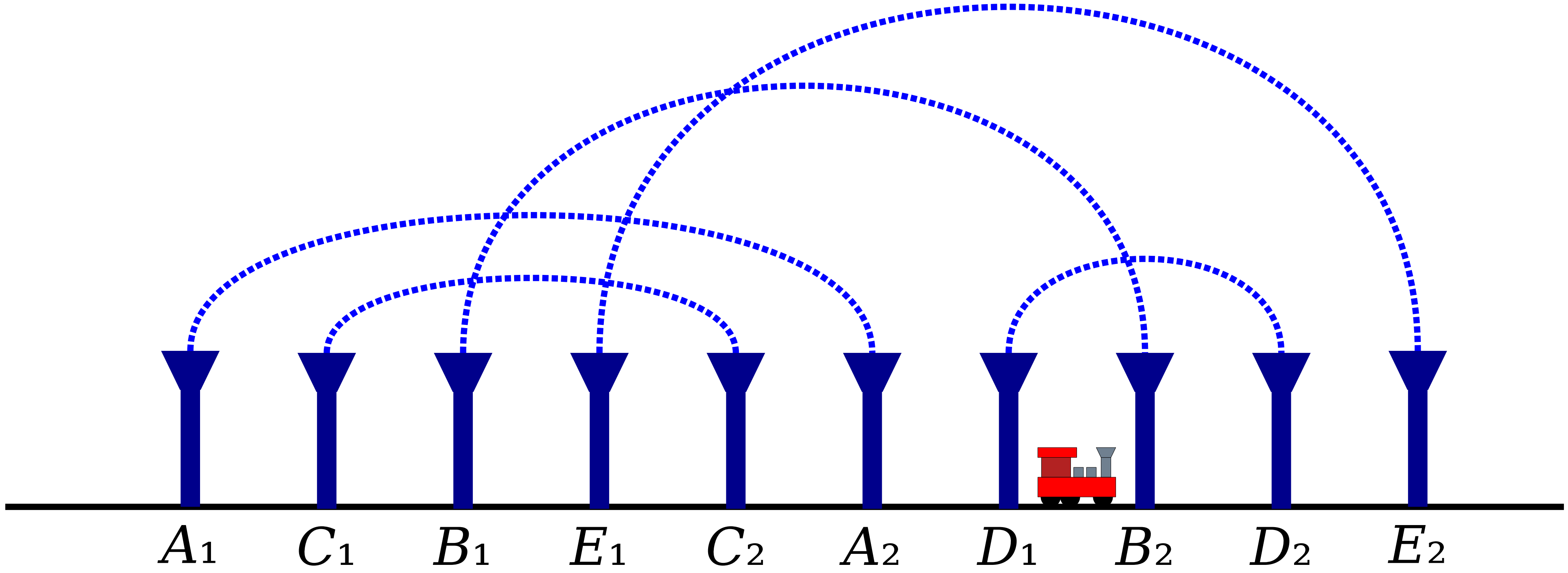
A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2

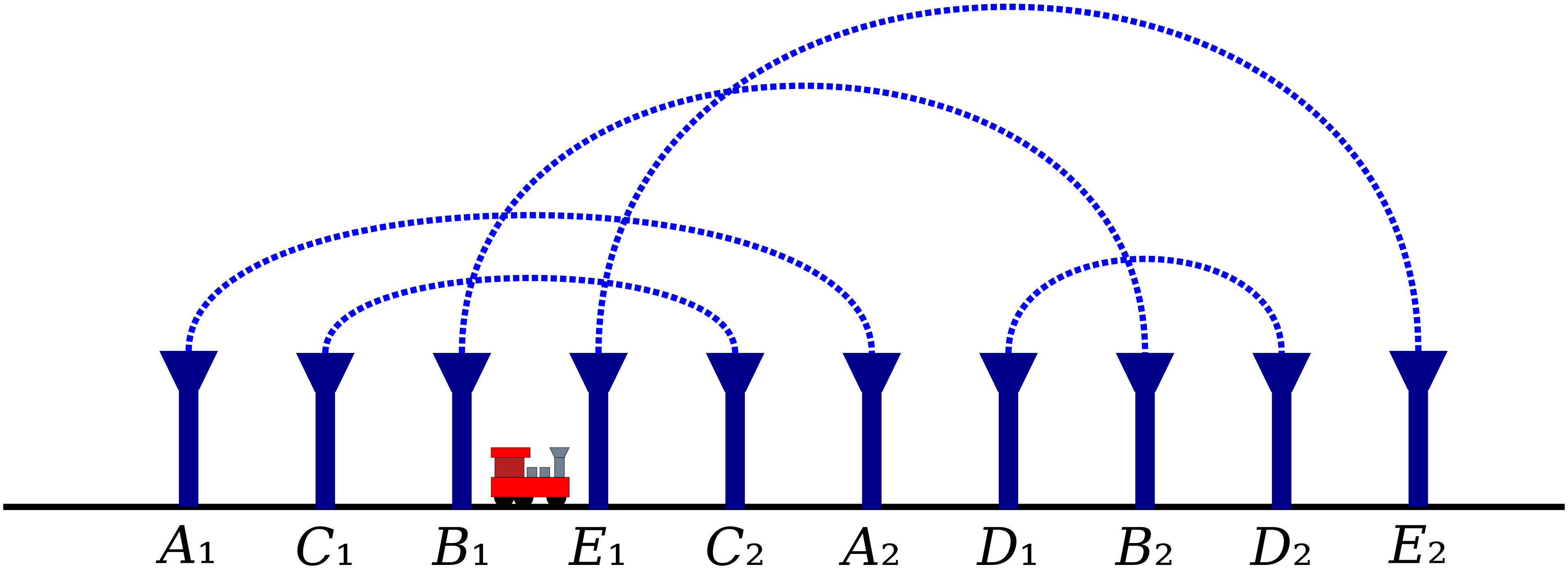


A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2

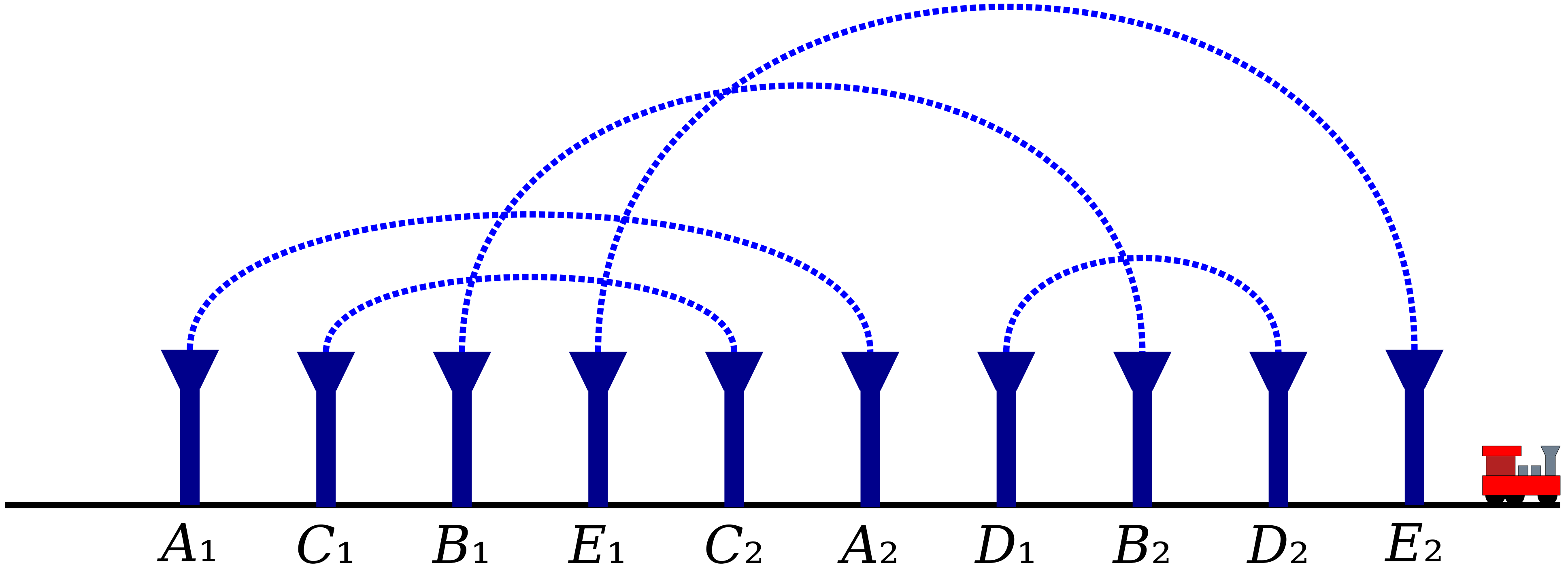


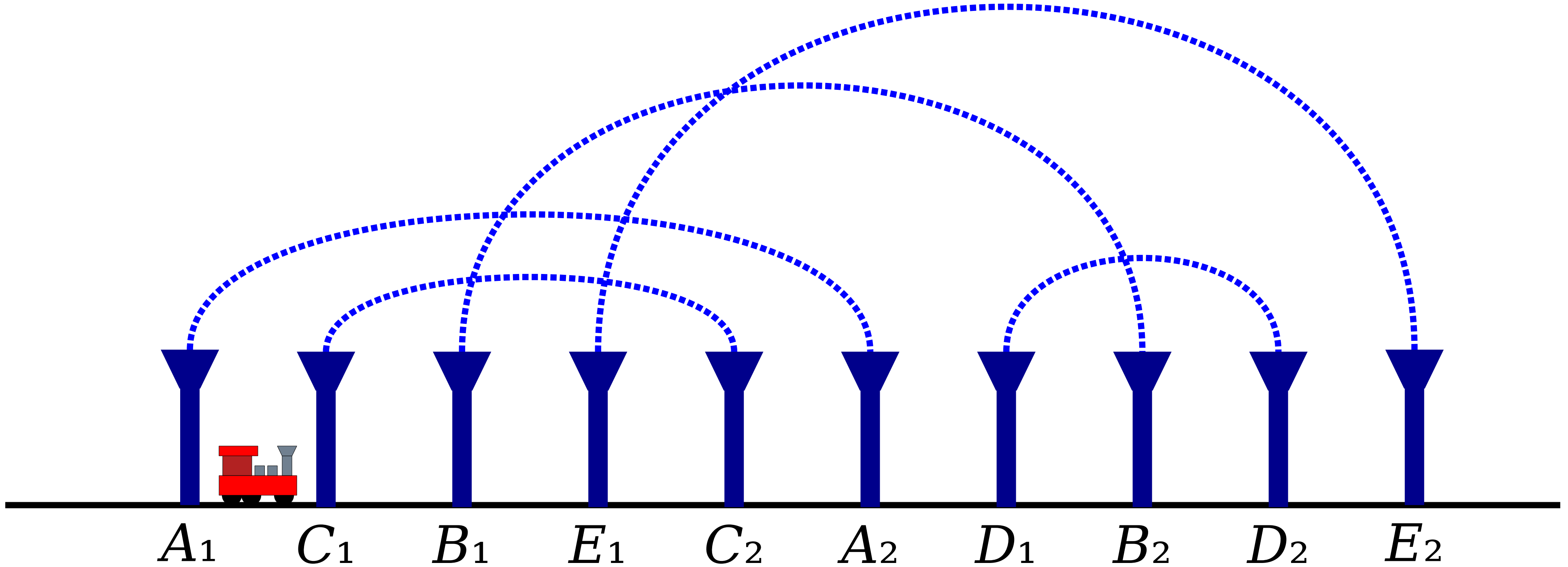
A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2


A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2

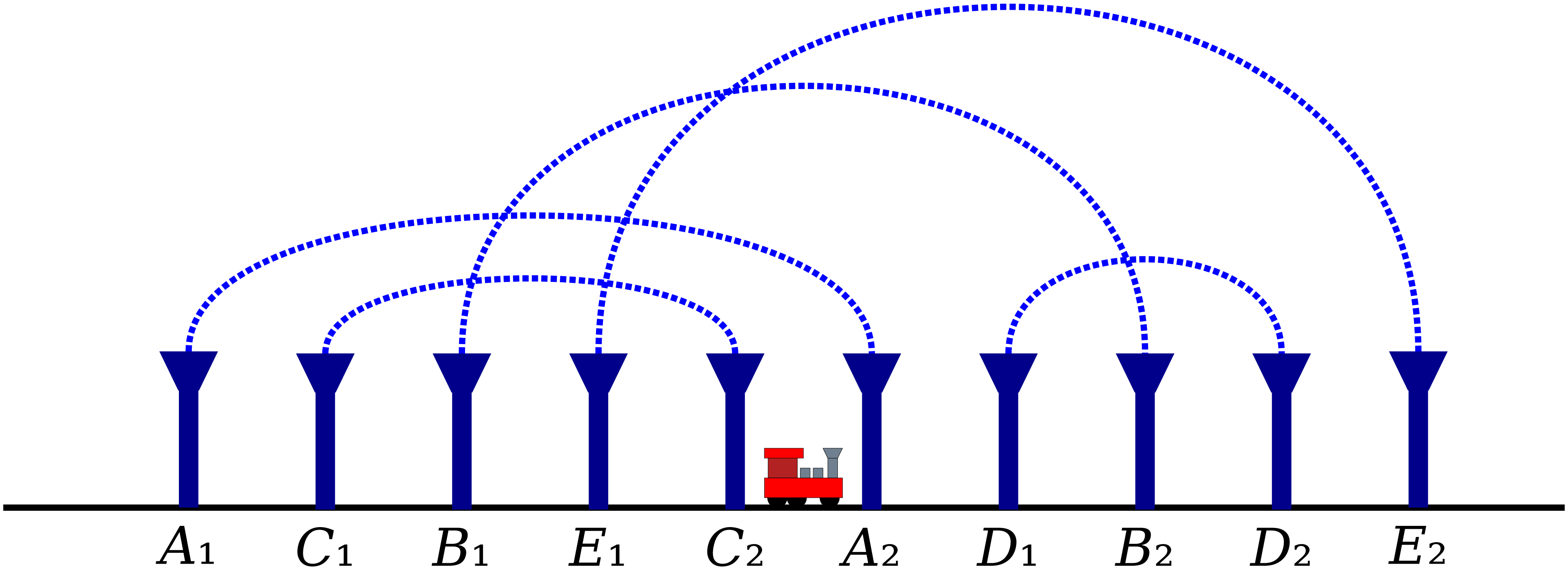


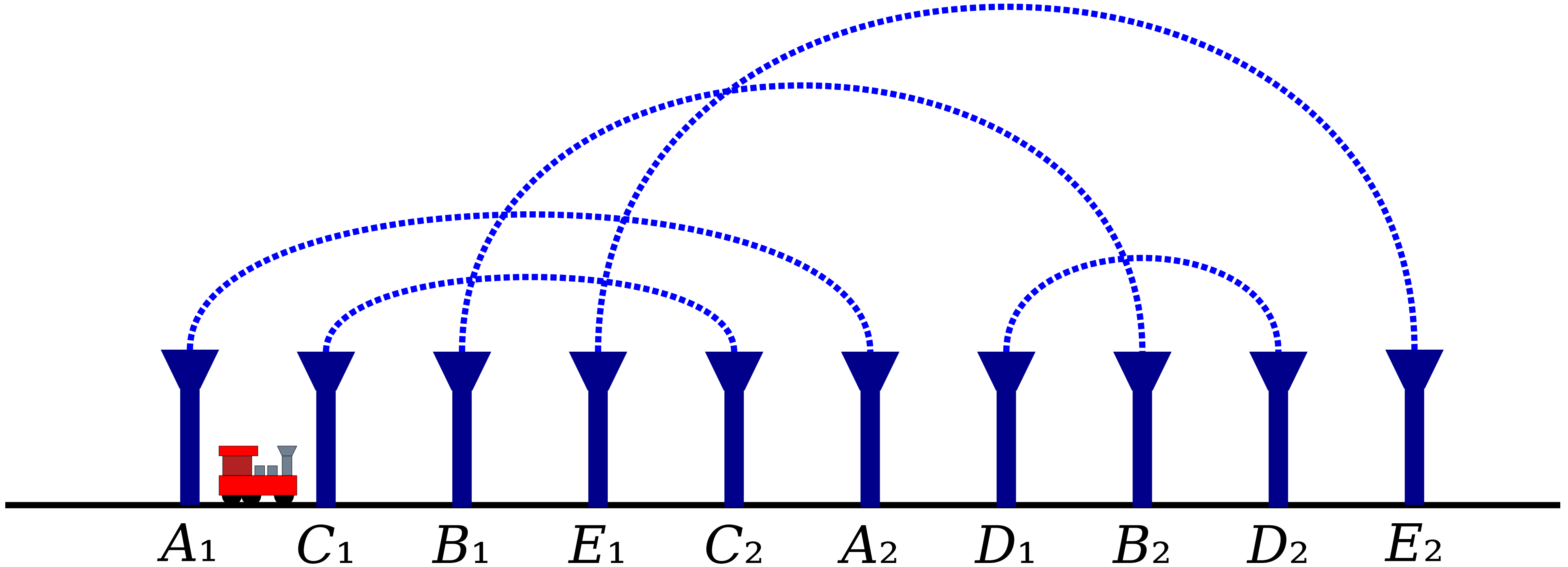
A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2

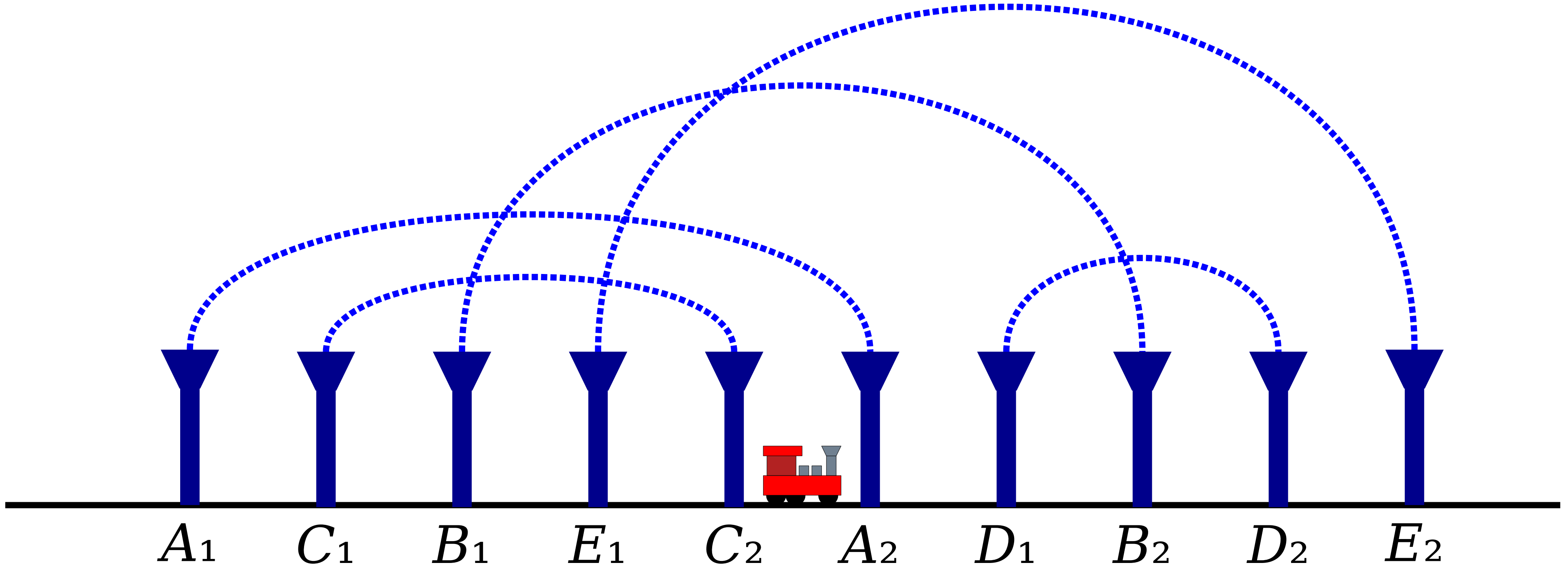

A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2



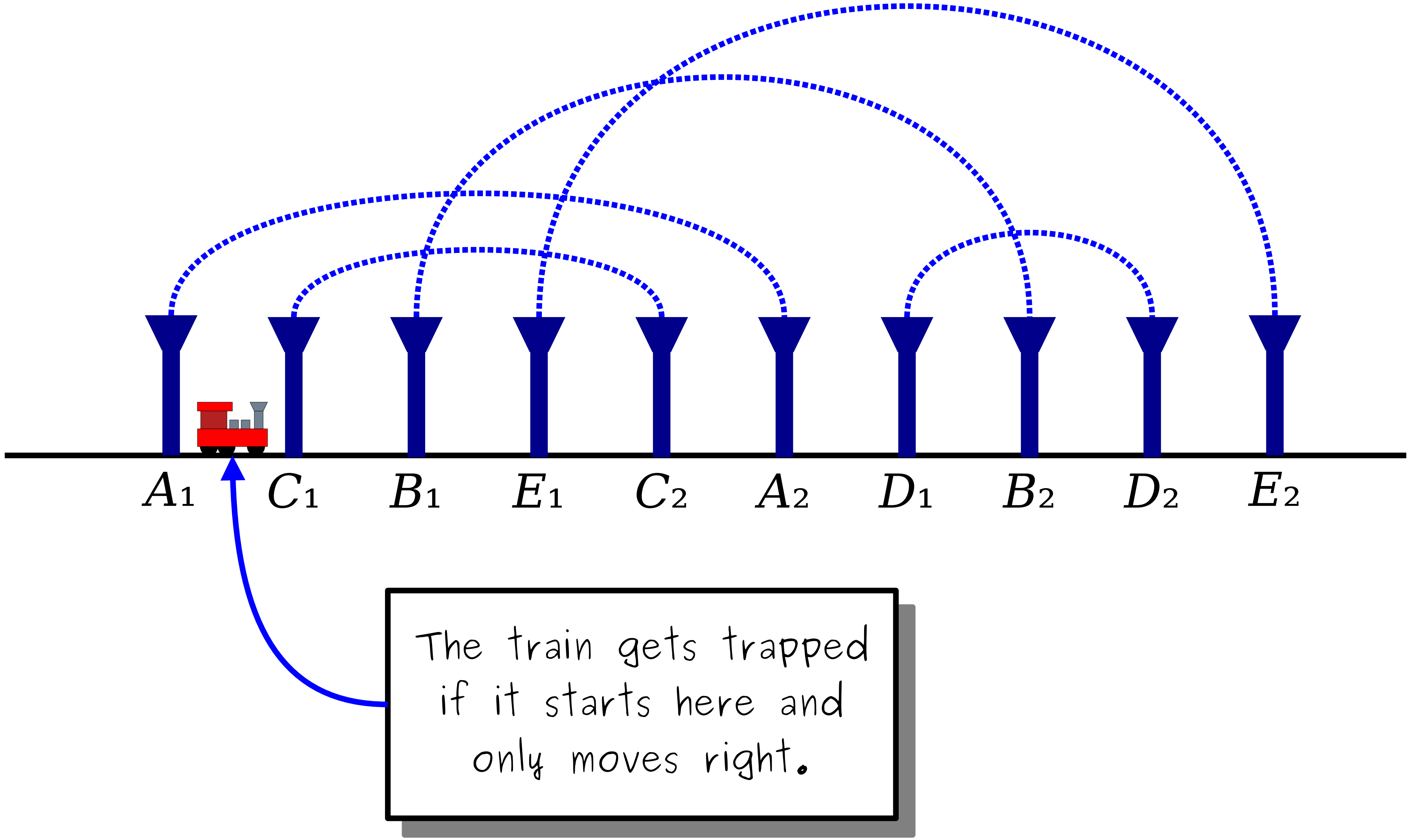
A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2


A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2


A_1
C_1
B_1
E_1
C_2
A_2
D_1
B_2
D_2
E_2



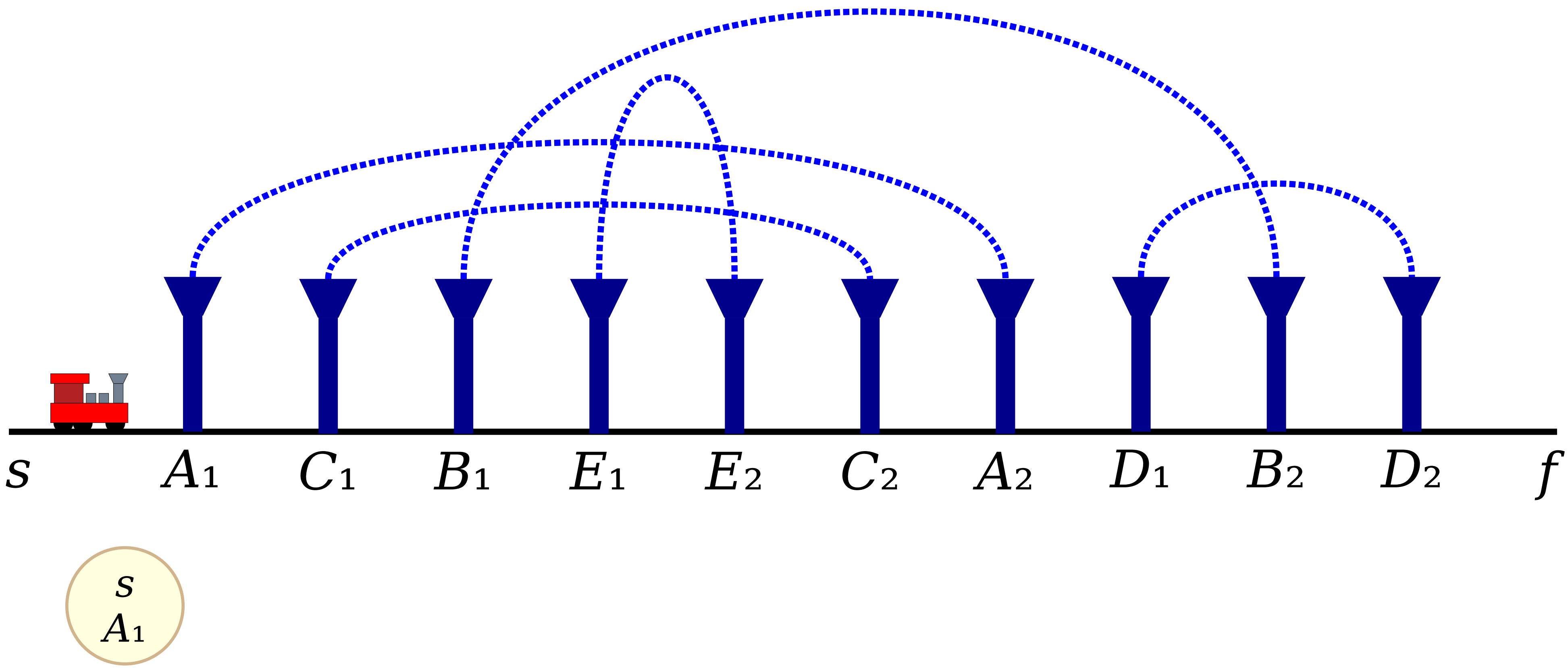
A1
C1
B1
E1
C2
A2
D1
B2
D2
E2

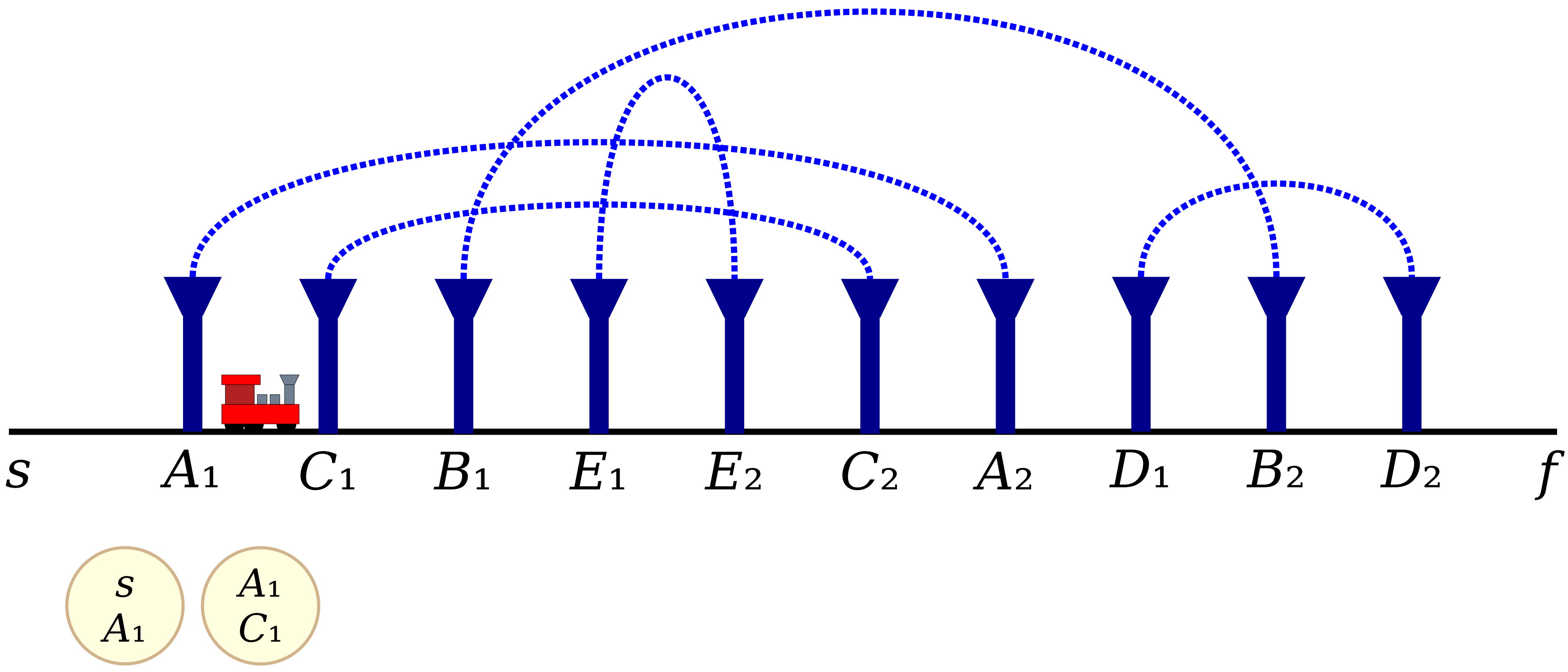


A_1 C_1 B_1 E_1 C_2 A_2 D_1 B_2 D_2 E_2
The train gets trapped if it starts here and only moves right.

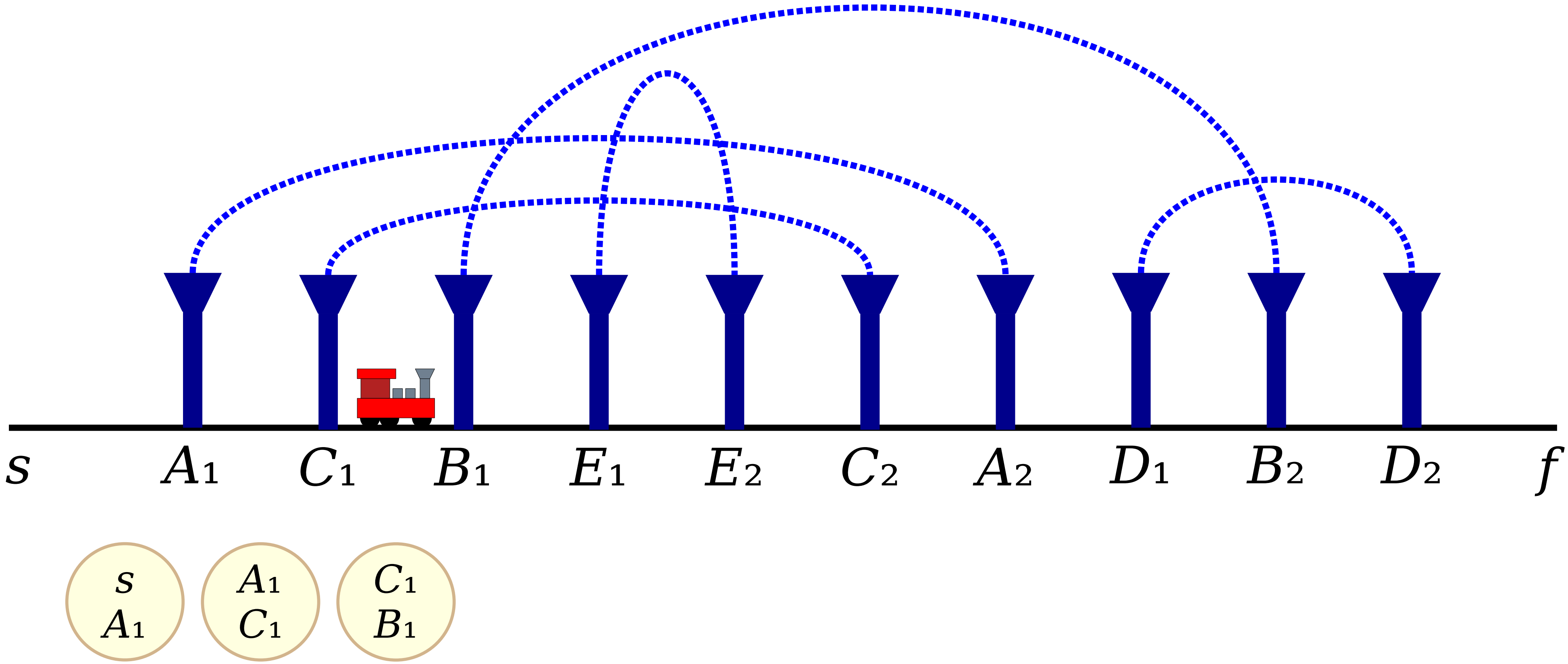
Can You Trap the Train?

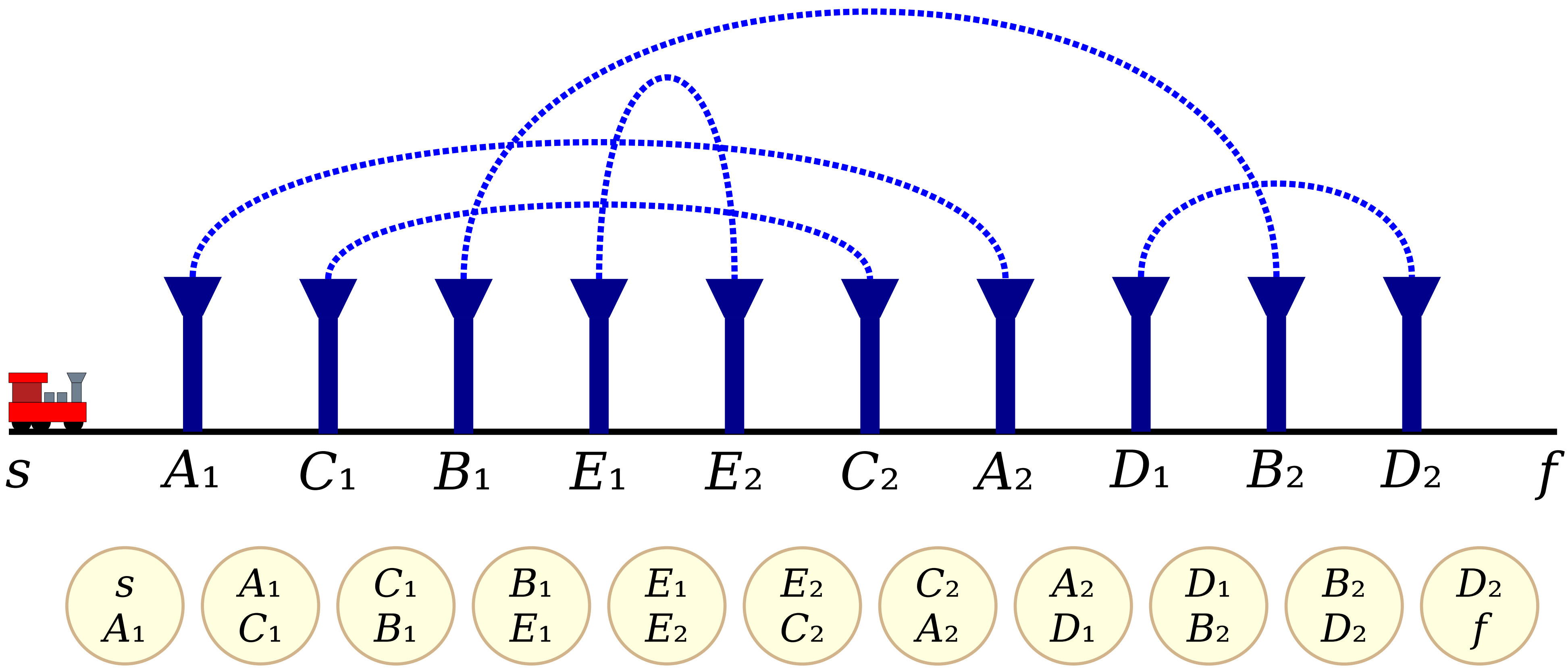
- The train always drives to the right.
- The train starts just before the first teleporter.
- Teleporters always link in pairs.
- Teleporters can’t stack on top of one another.
- Teleporters can’t appear at or after the end point.
- You can use as many teleporters as you’d like.

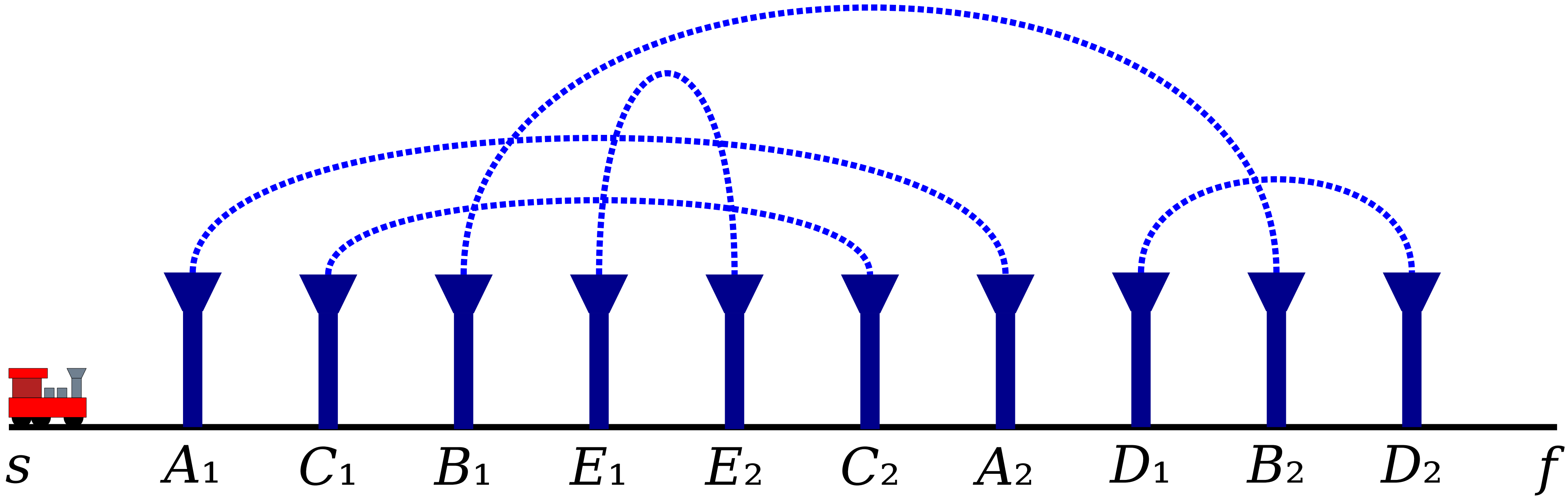

s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s
A1

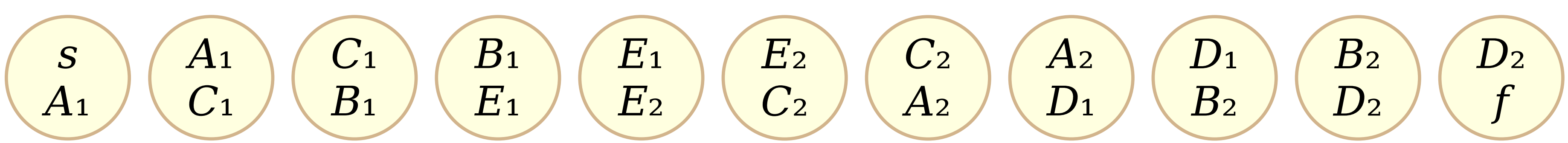


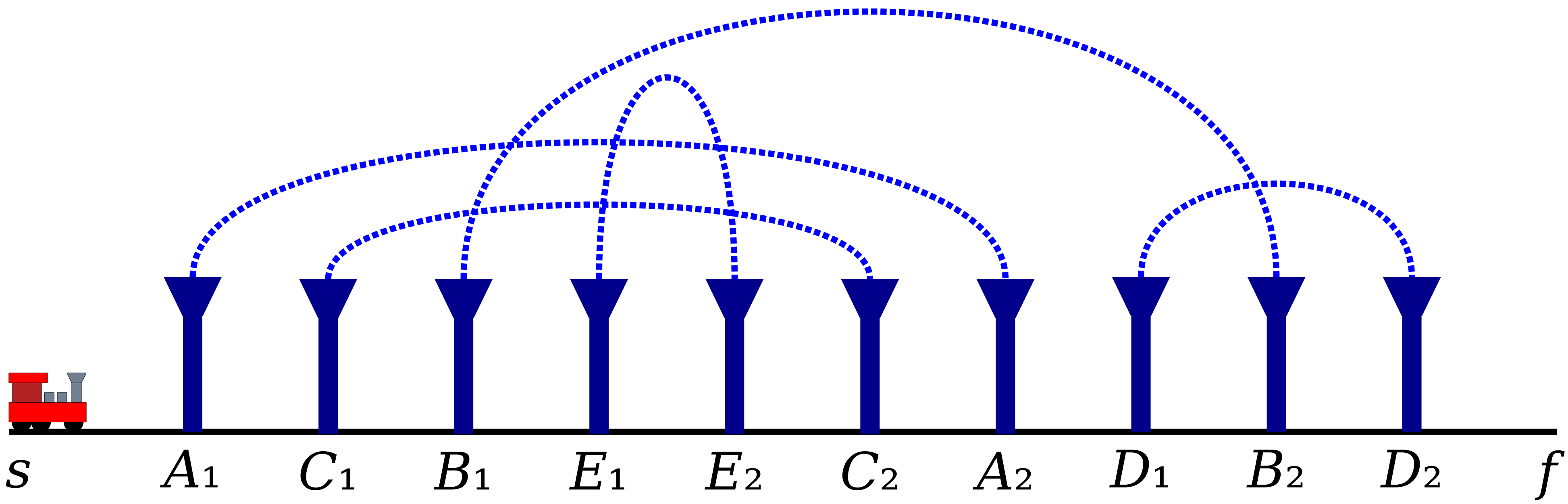
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s
A1
A1
C1

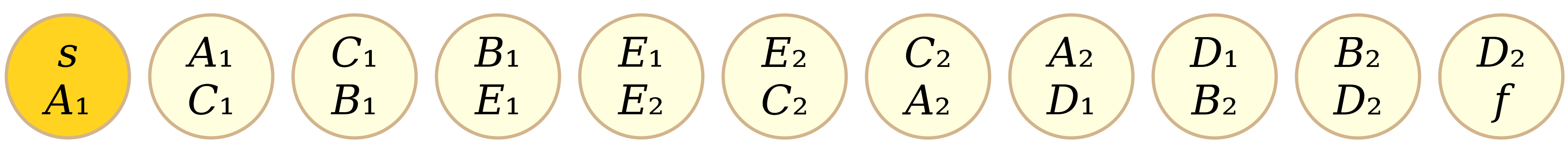

s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A1 C1
C1 B1

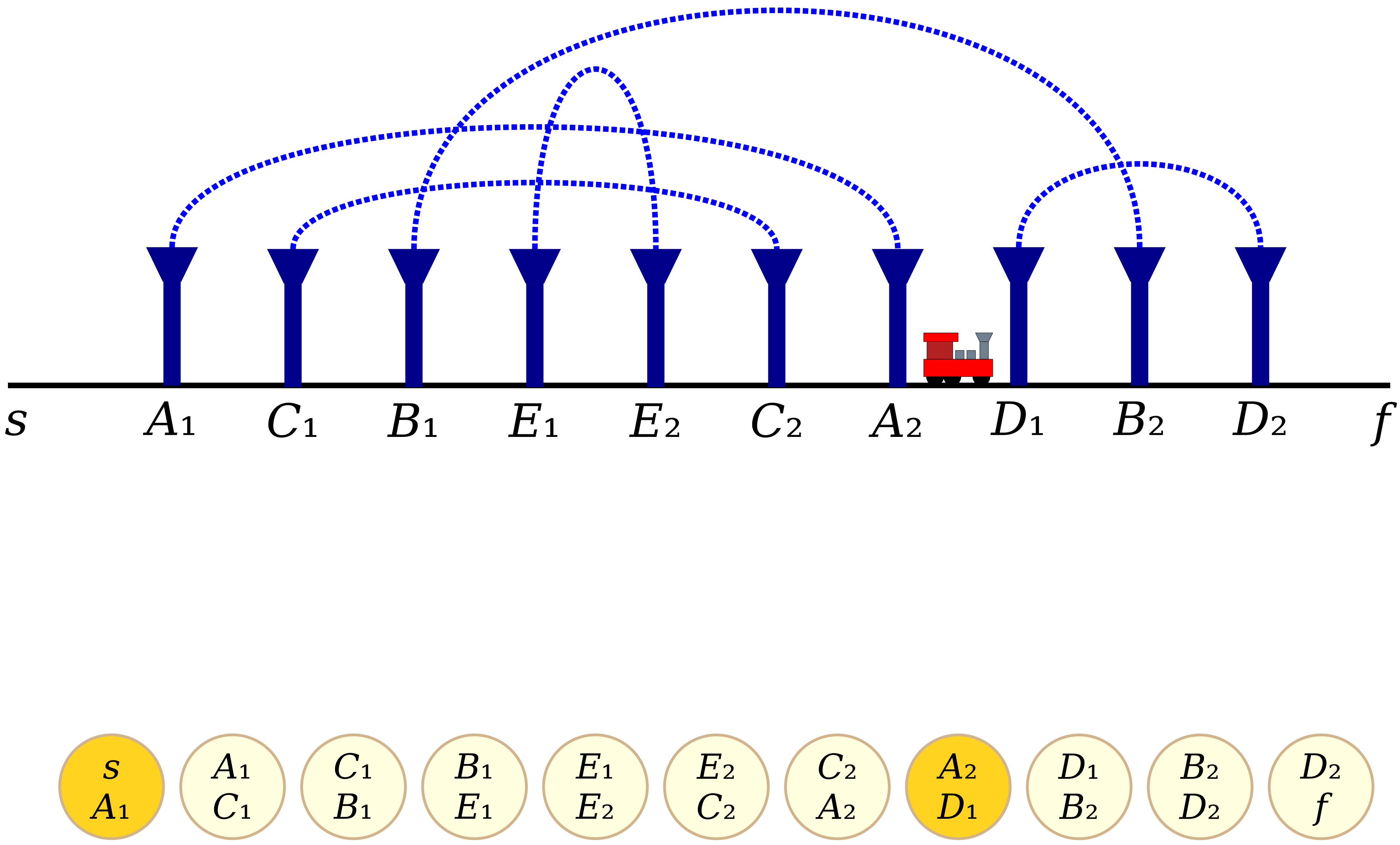

s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
A2 D1
D1 B2
B2 D2
D2 f


s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f

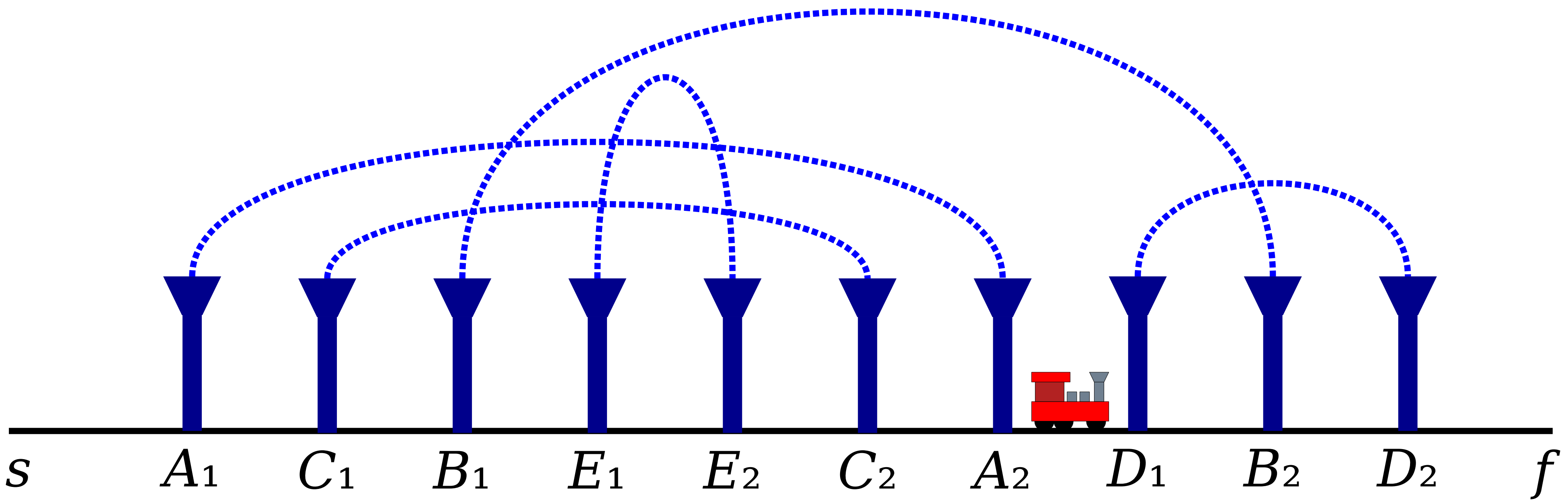

s A_1
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
A_2 D_1
D_1 B_2
B_2 D_2
D_2 f

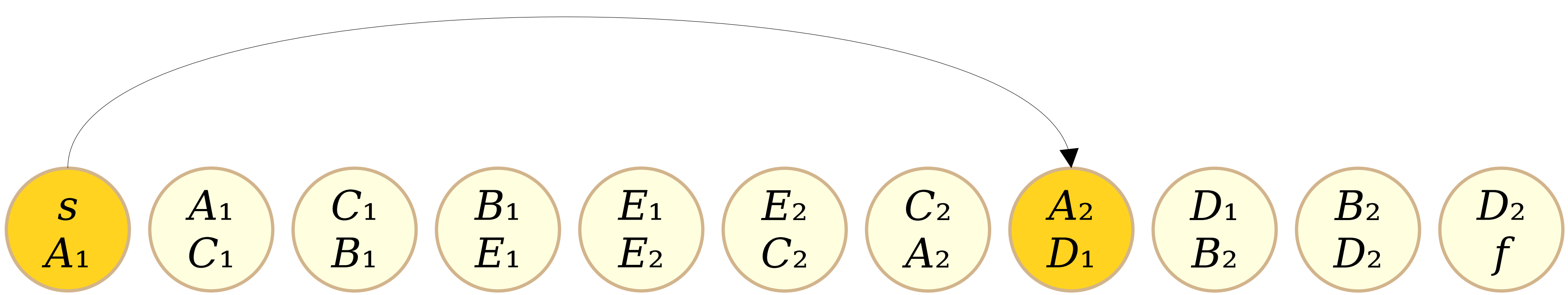

s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f

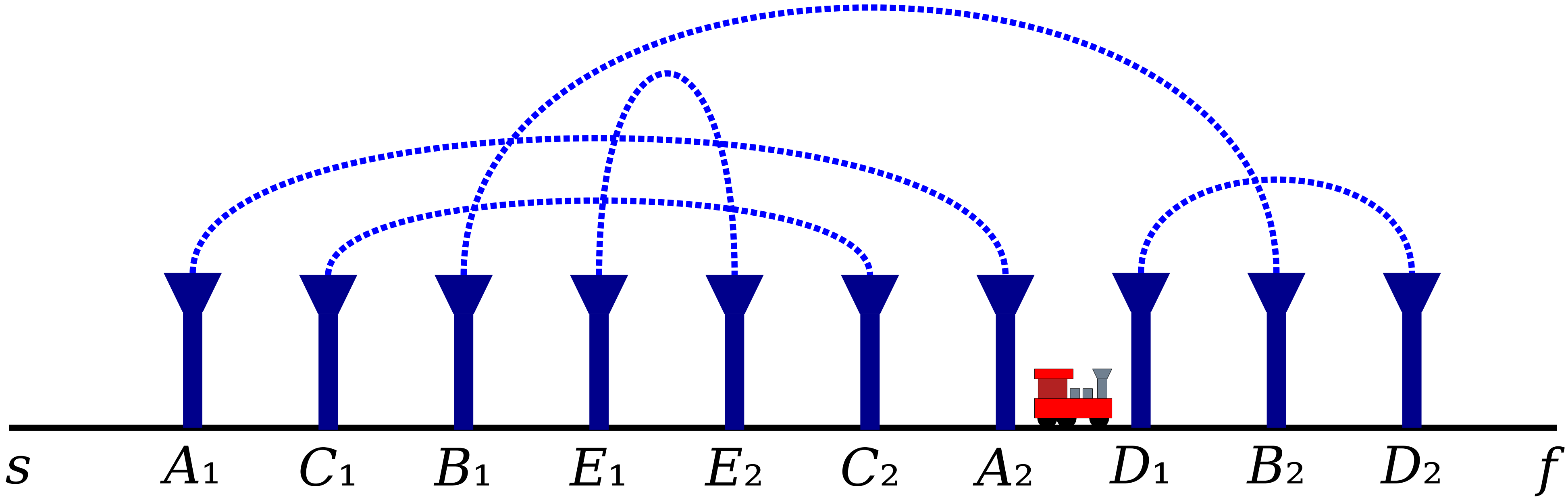

s A1
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
A2 D1
D1 B2
B2 D2
D2 f

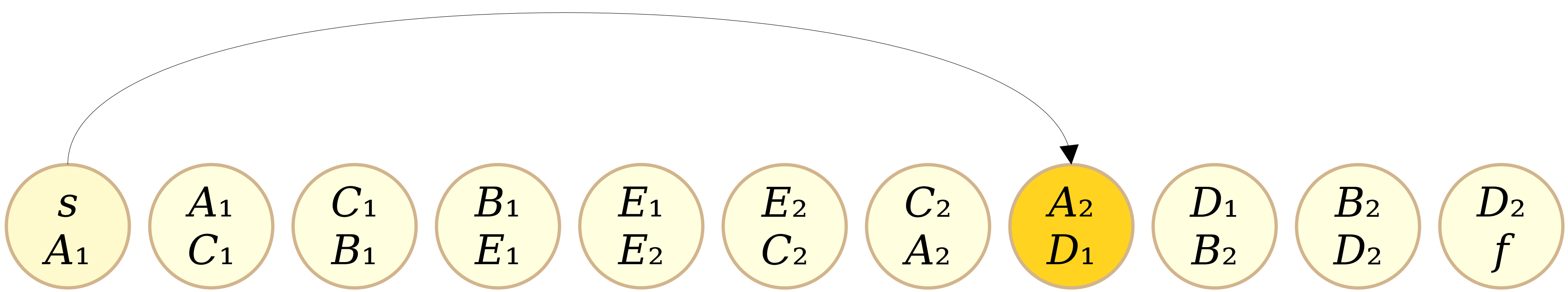


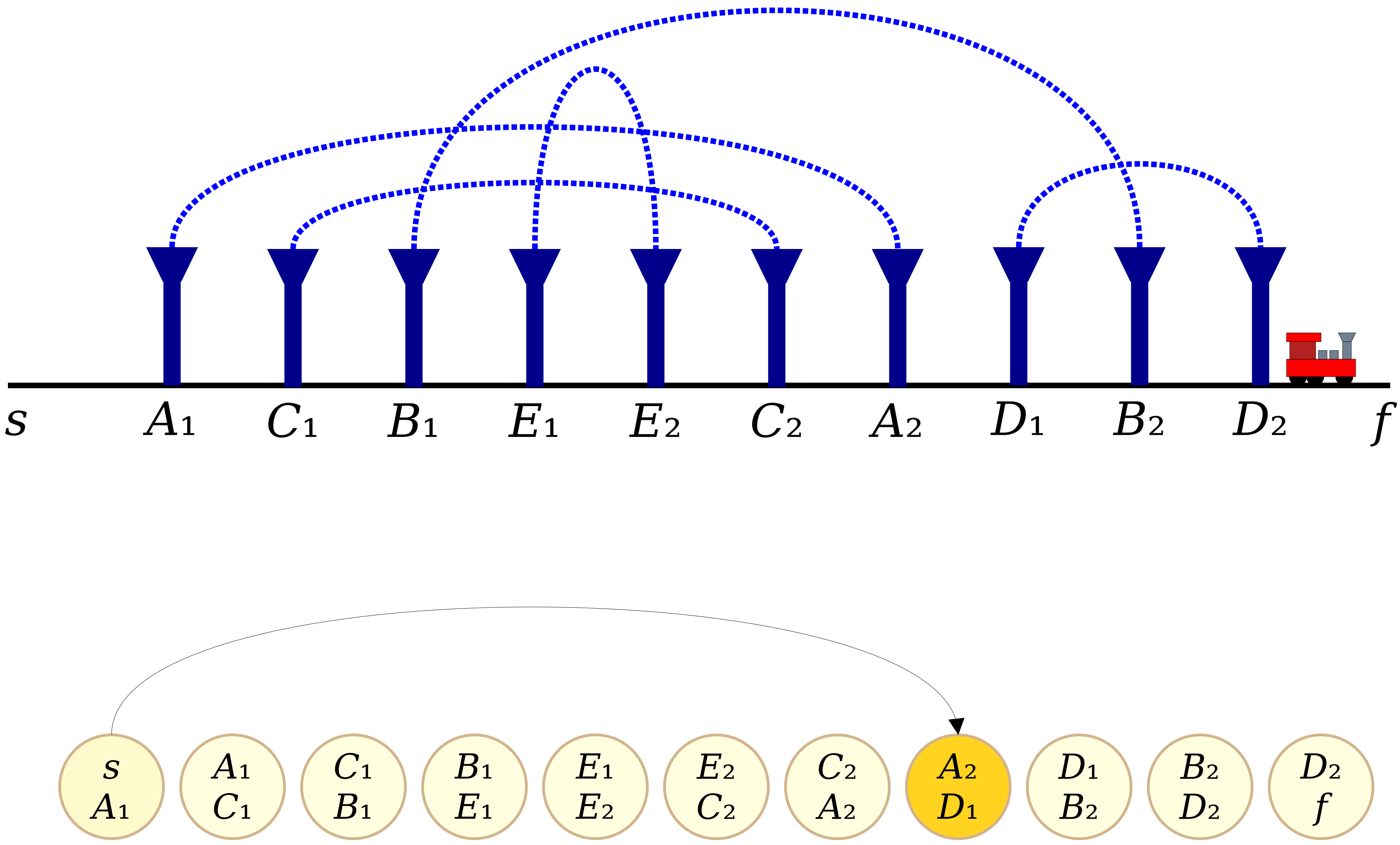
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
A2 D1
D1 B2
B2 D2
D2 f


s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f

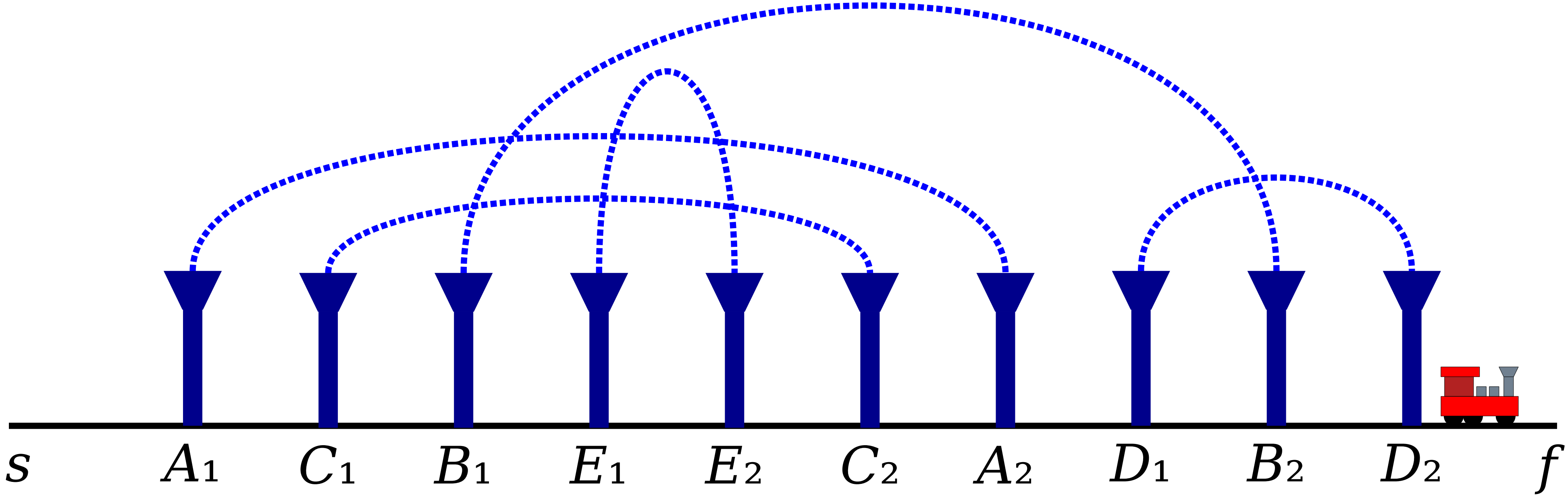

s A_1
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
A_2 D_1
D_1 B_2
B_2 D_2
D_2 f

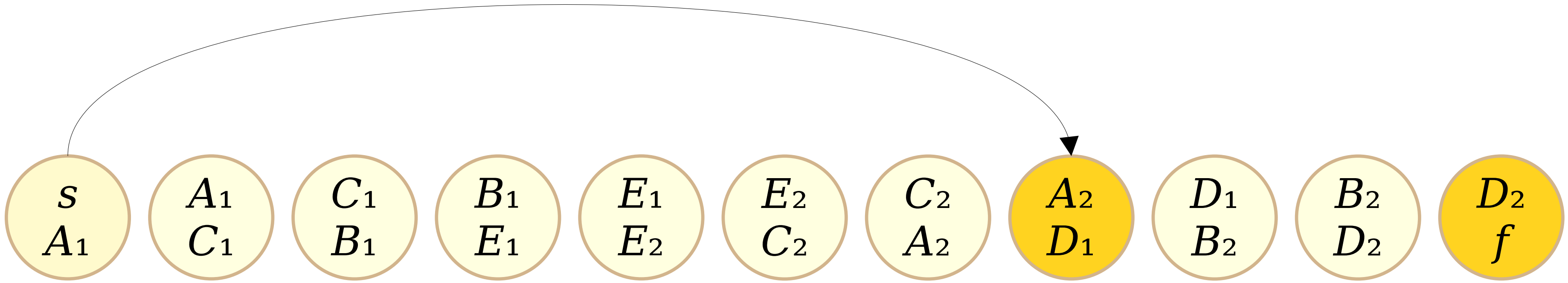

s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f

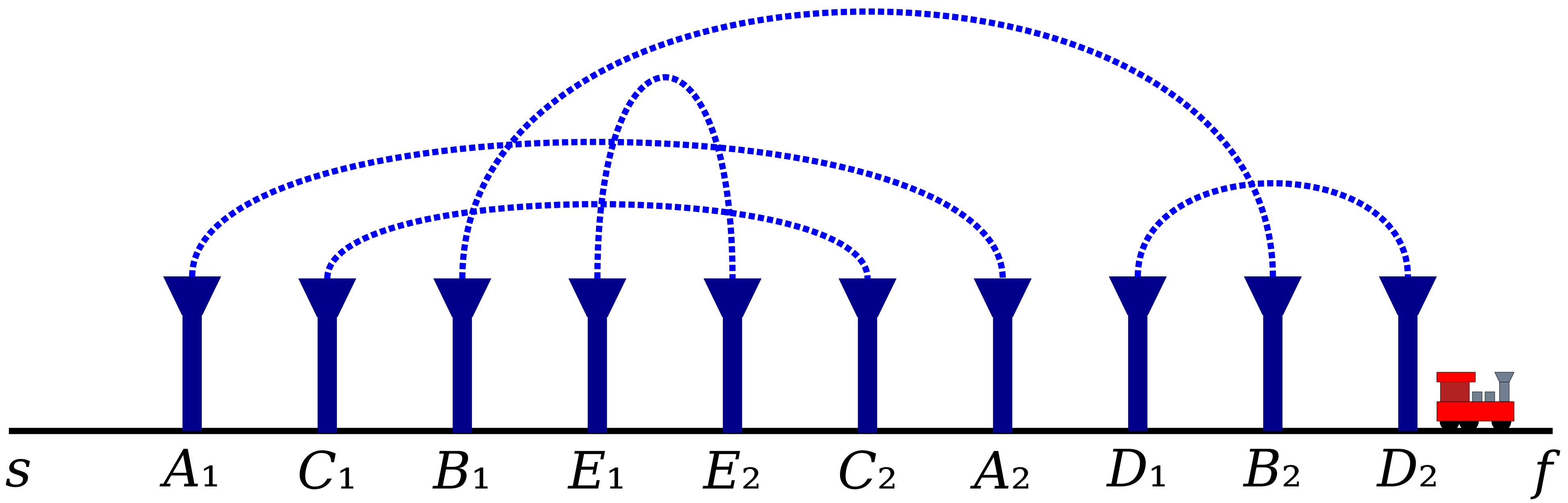

s A1
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
A2 D1
D1 B2
B2 D2
D2 f



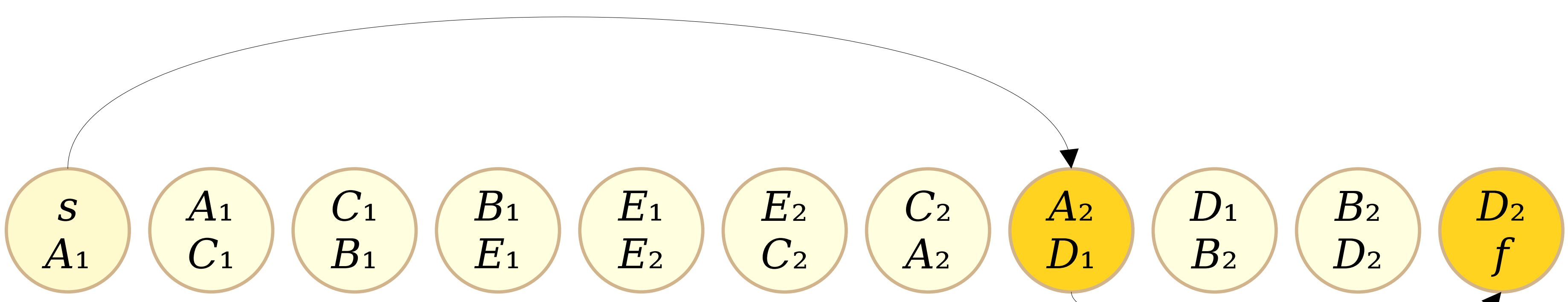
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
A2 D1
D1 B2
B2 D2
D2 f


s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f

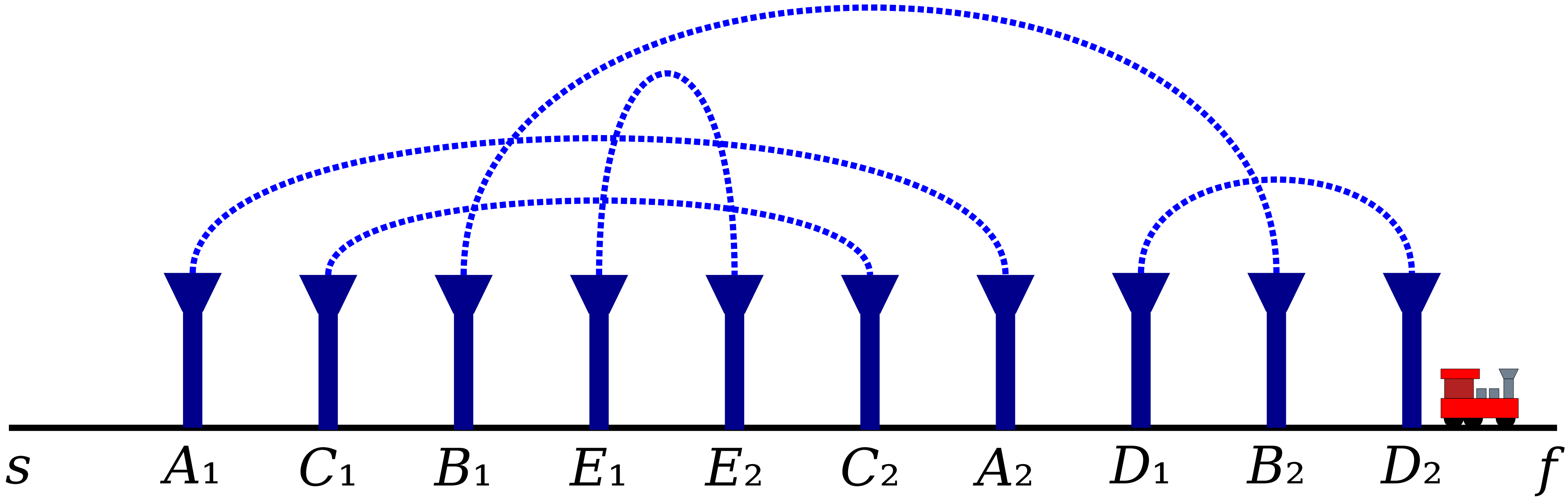

s A_1
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
A_2 D_1
D_1 B_2
B_2 D_2
D_2 f

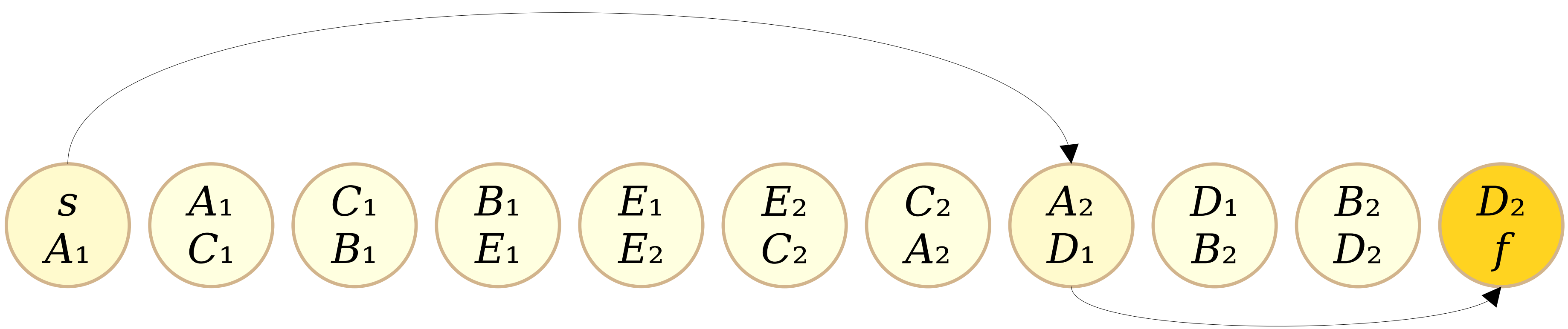


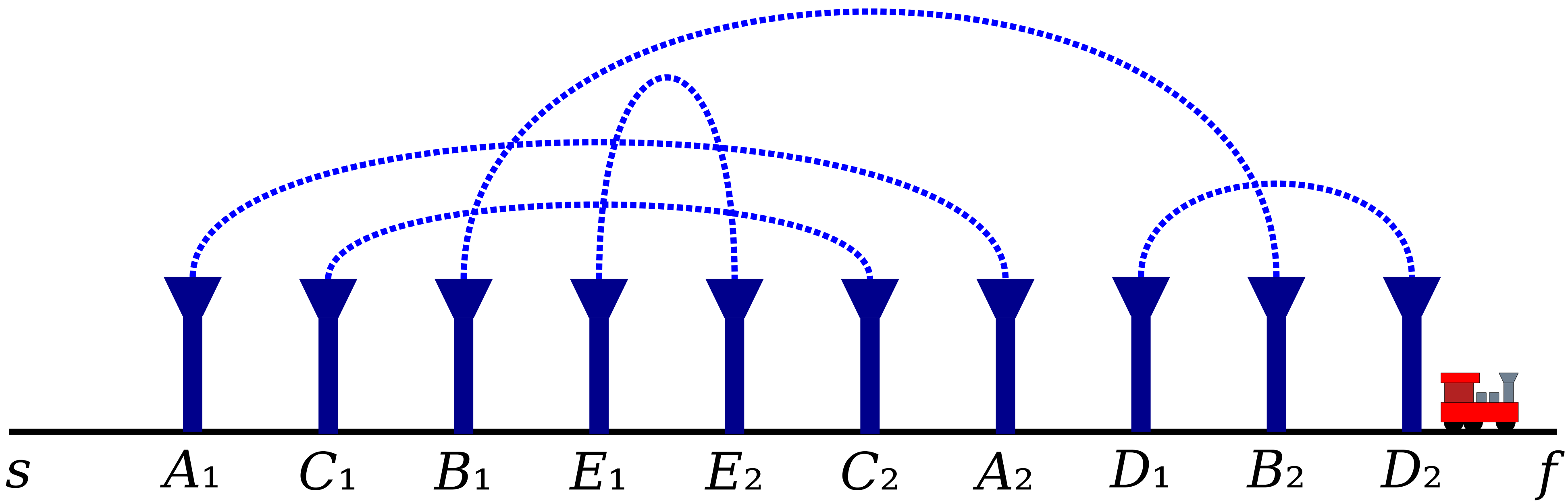
s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f

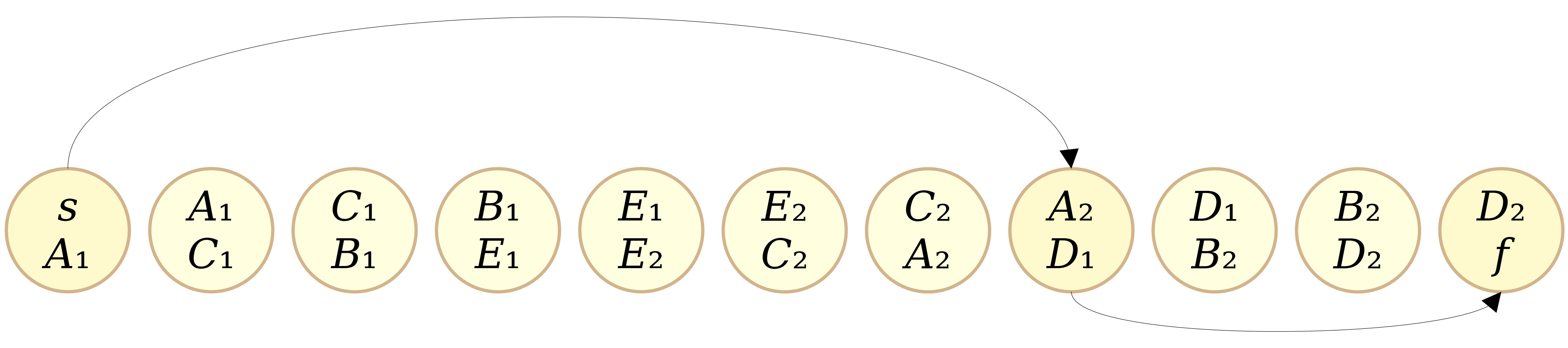


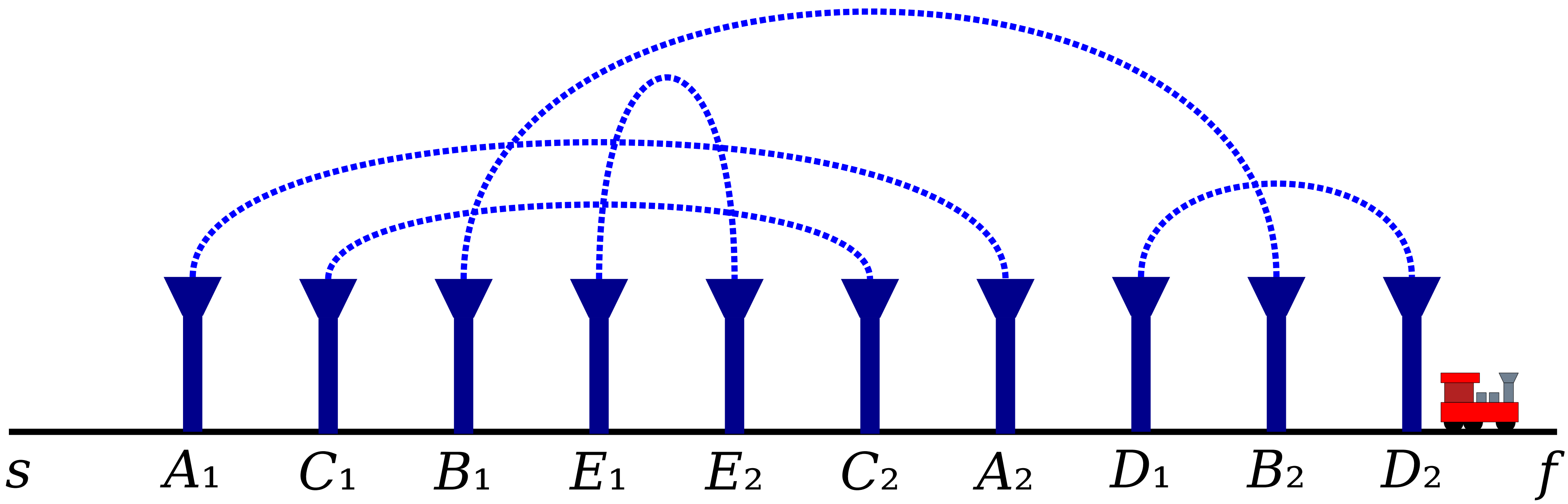
s A_1
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
A_2 D_1
D_1 B_2
B_2 D_2
D_2 f


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f

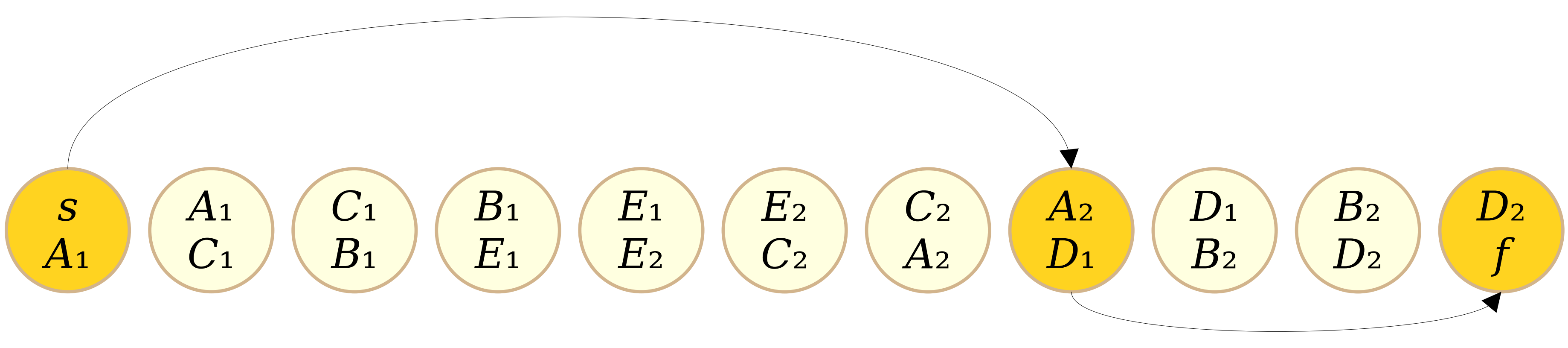

s A1
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
A2 D1
D1 B2
B2 D2
D2 f


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f

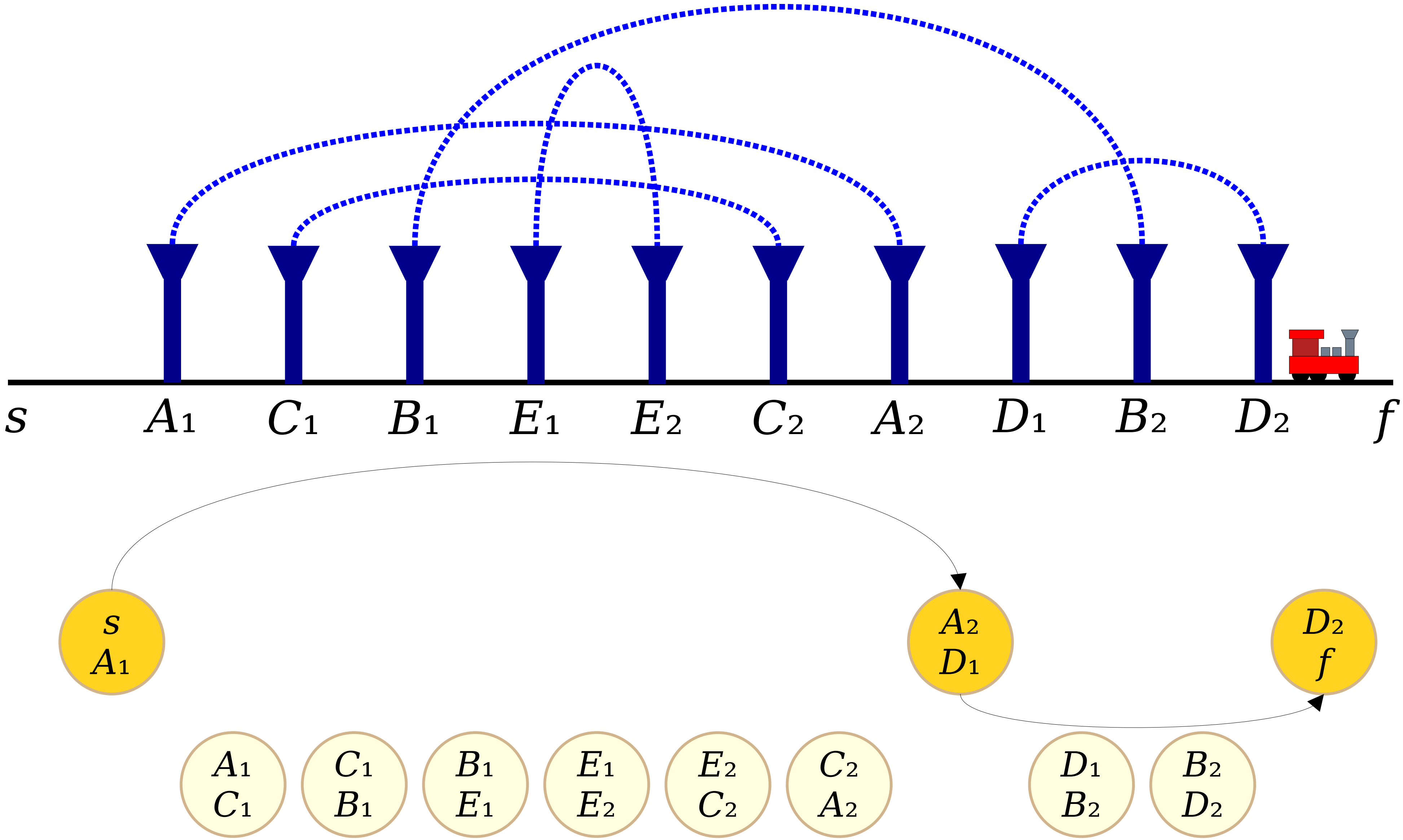

s A1
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
A2 D1
D1 B2
B2 D2
D2 f



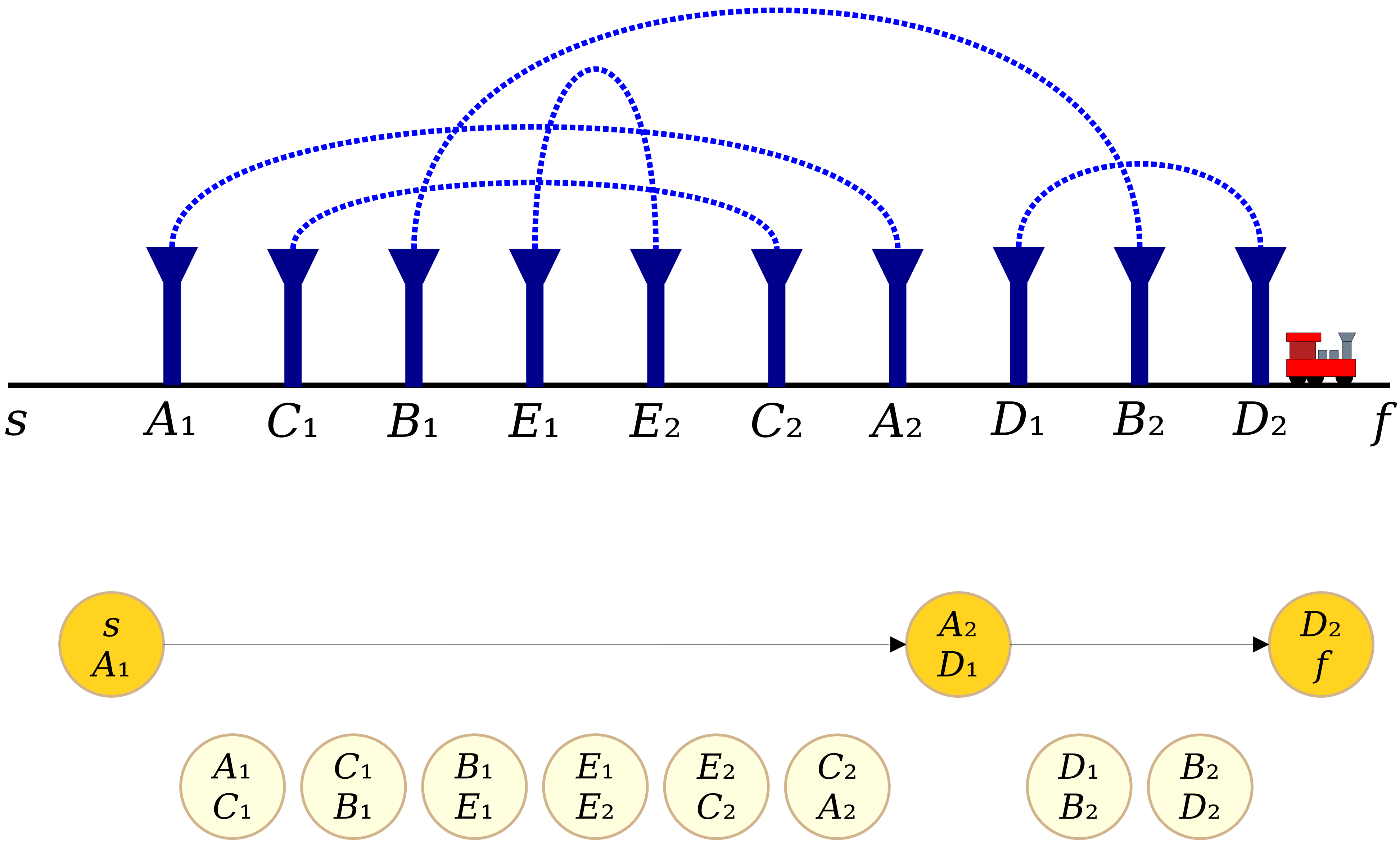
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f

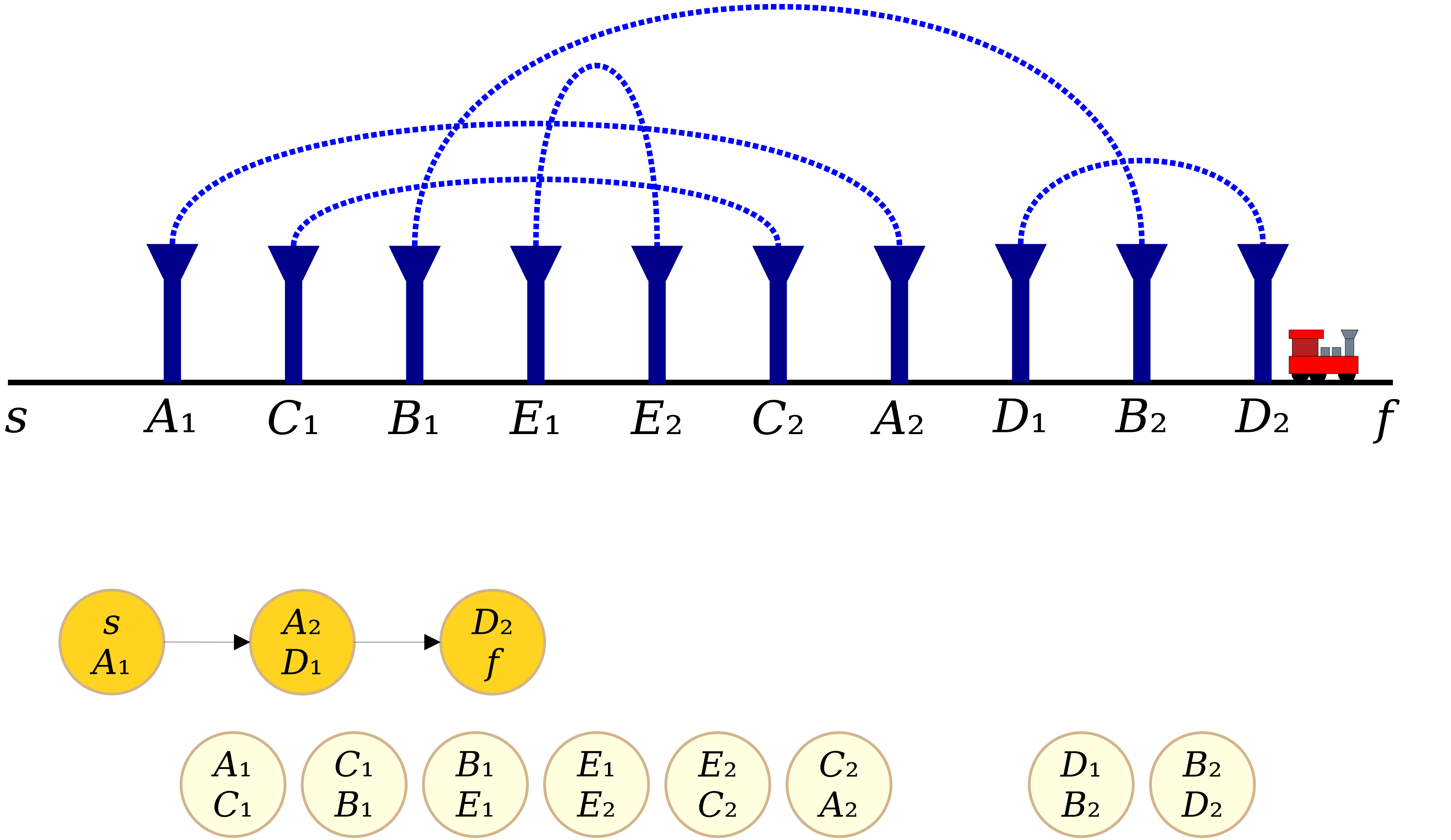


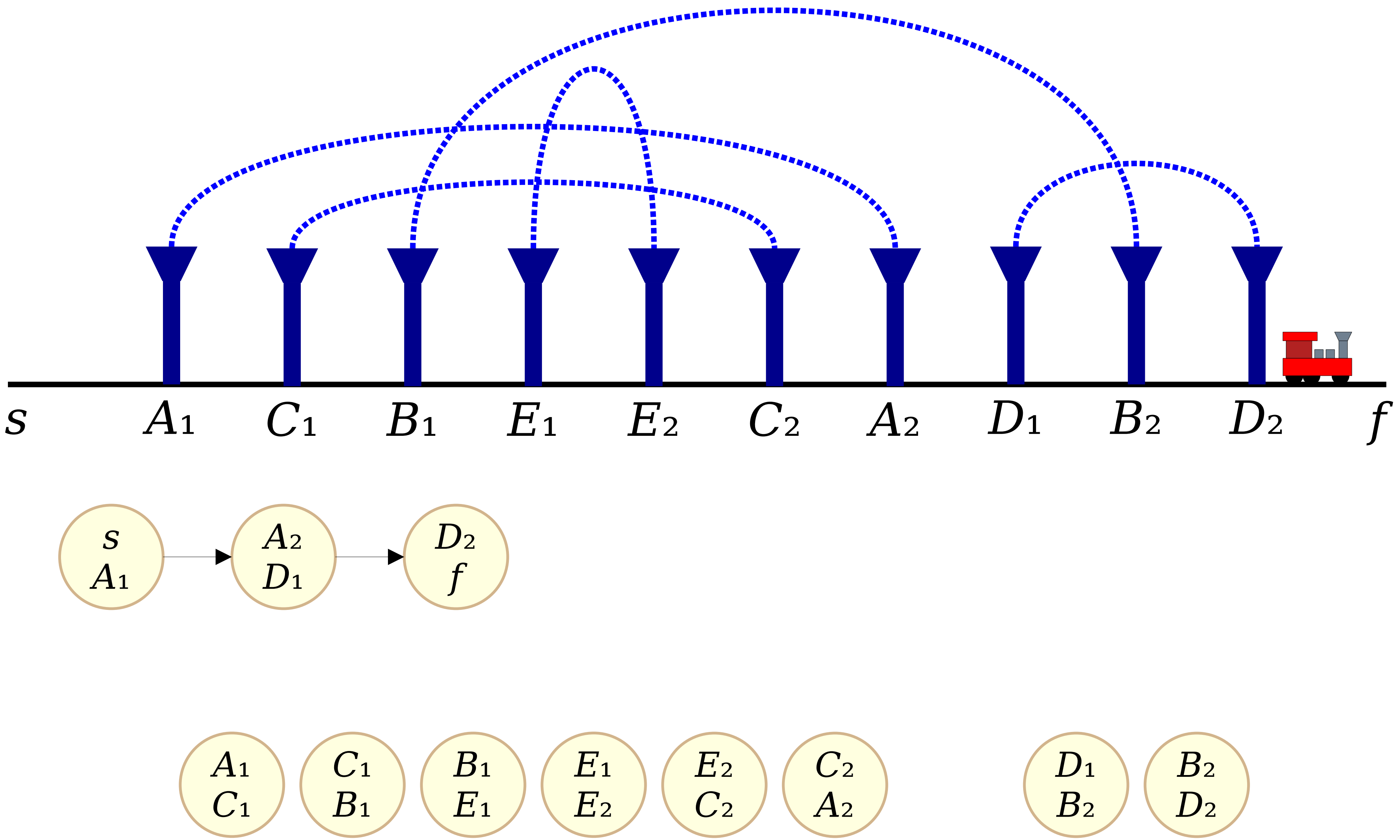
s A1
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
A2 D1
D1 B2
B2 D2
D2 f



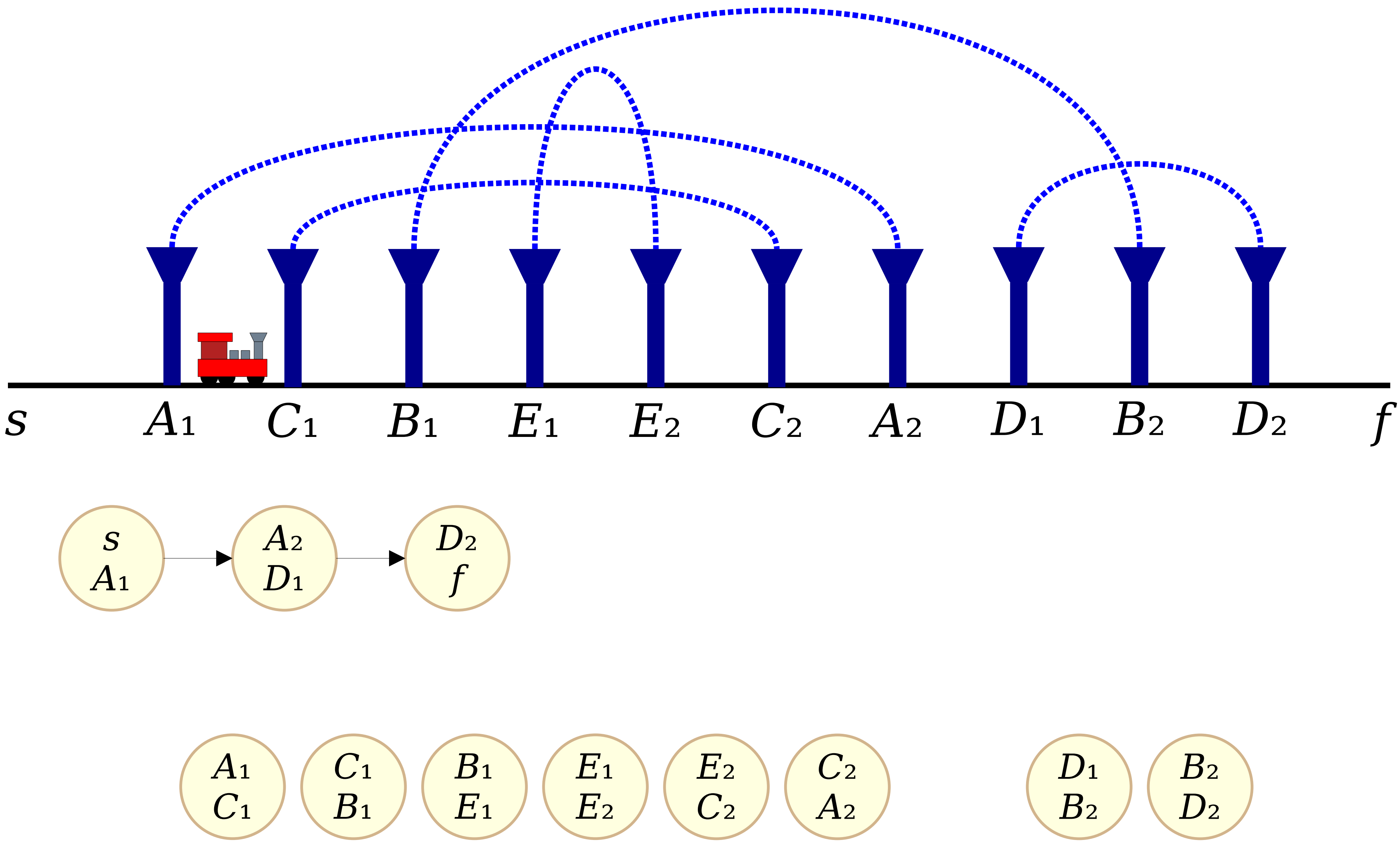
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f


s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
D_1 B_2
B_2 D_2

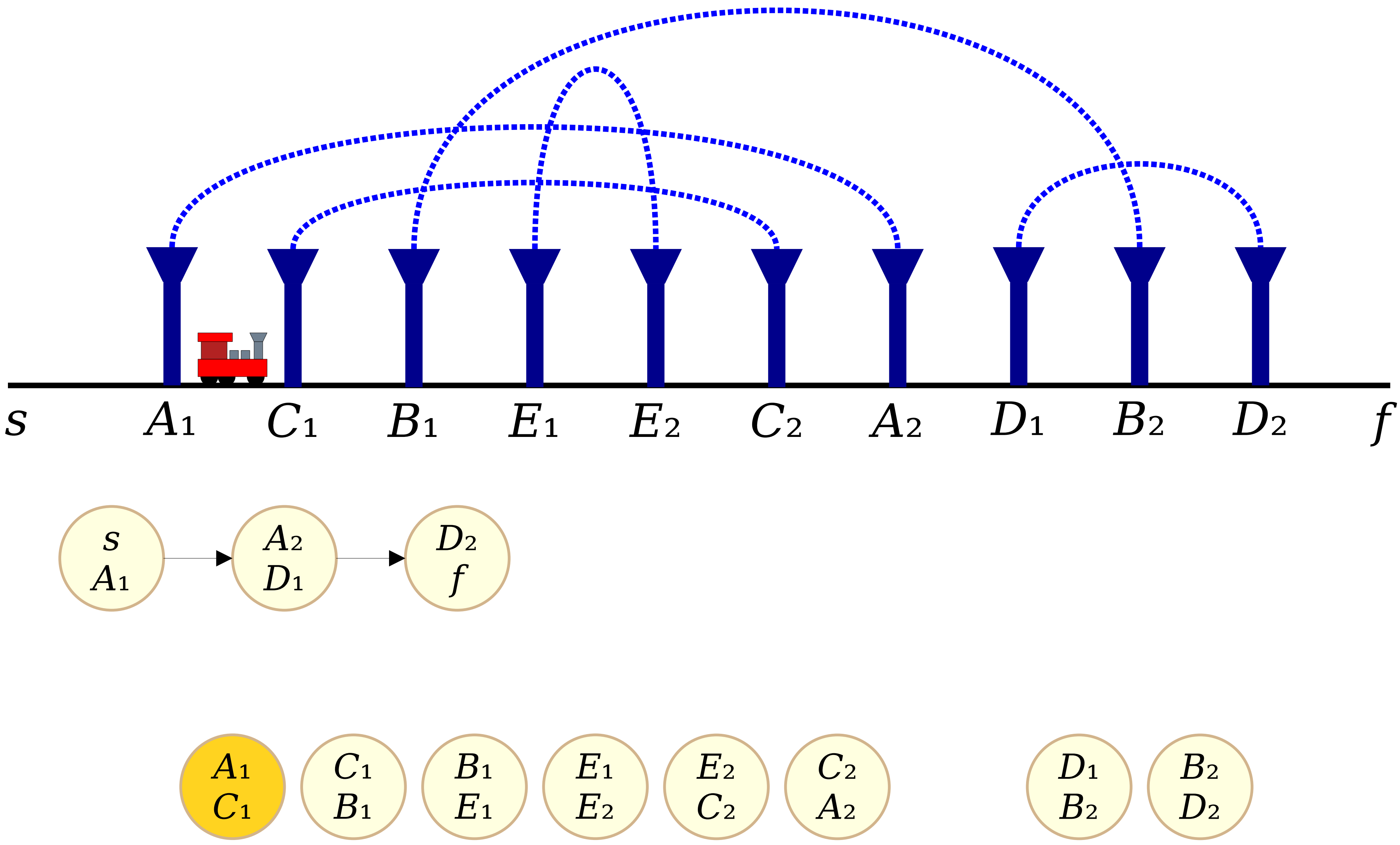

s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
D_1 B_2
B_2 D_2



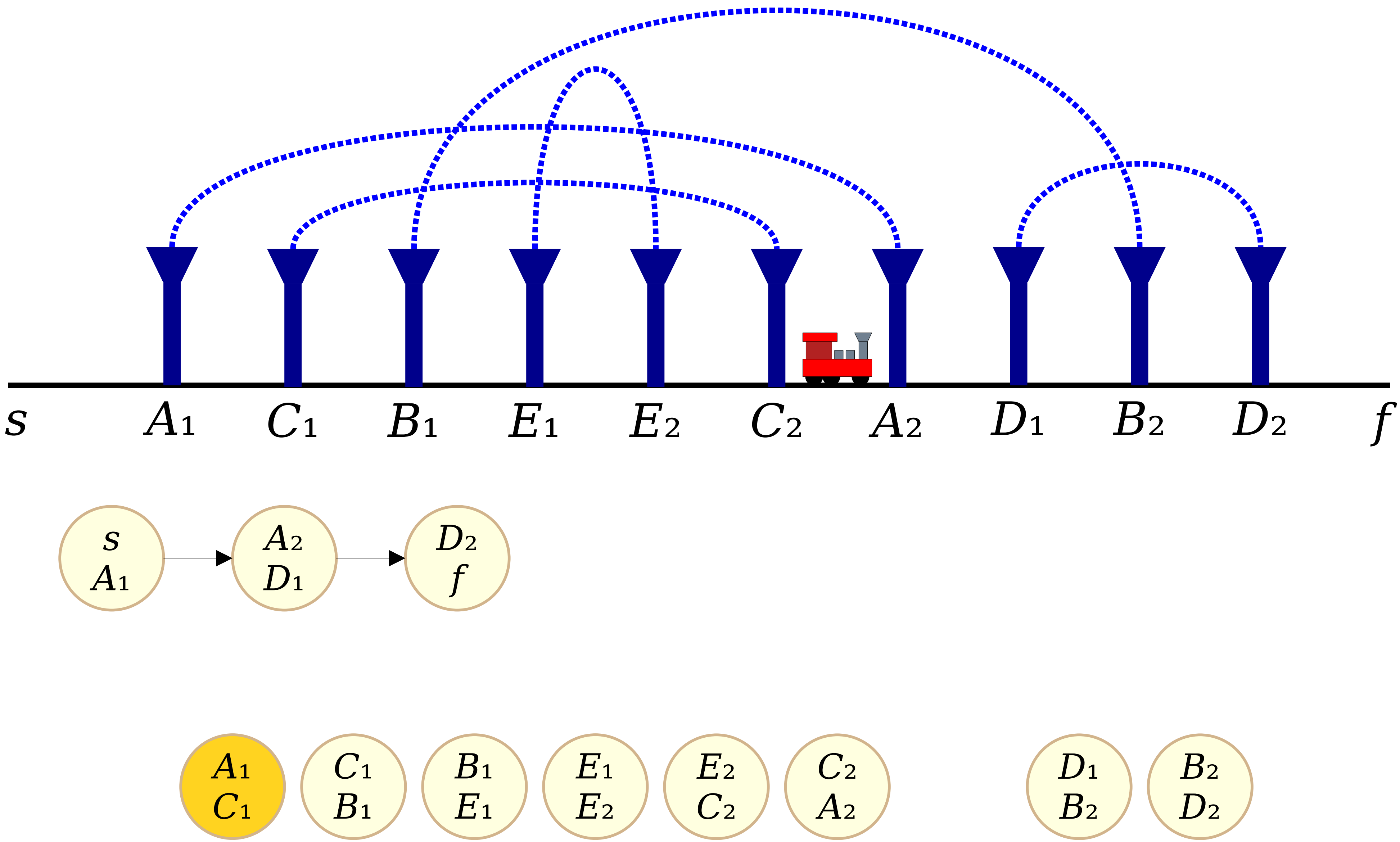
s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
D_1 B_2
B_2 D_2



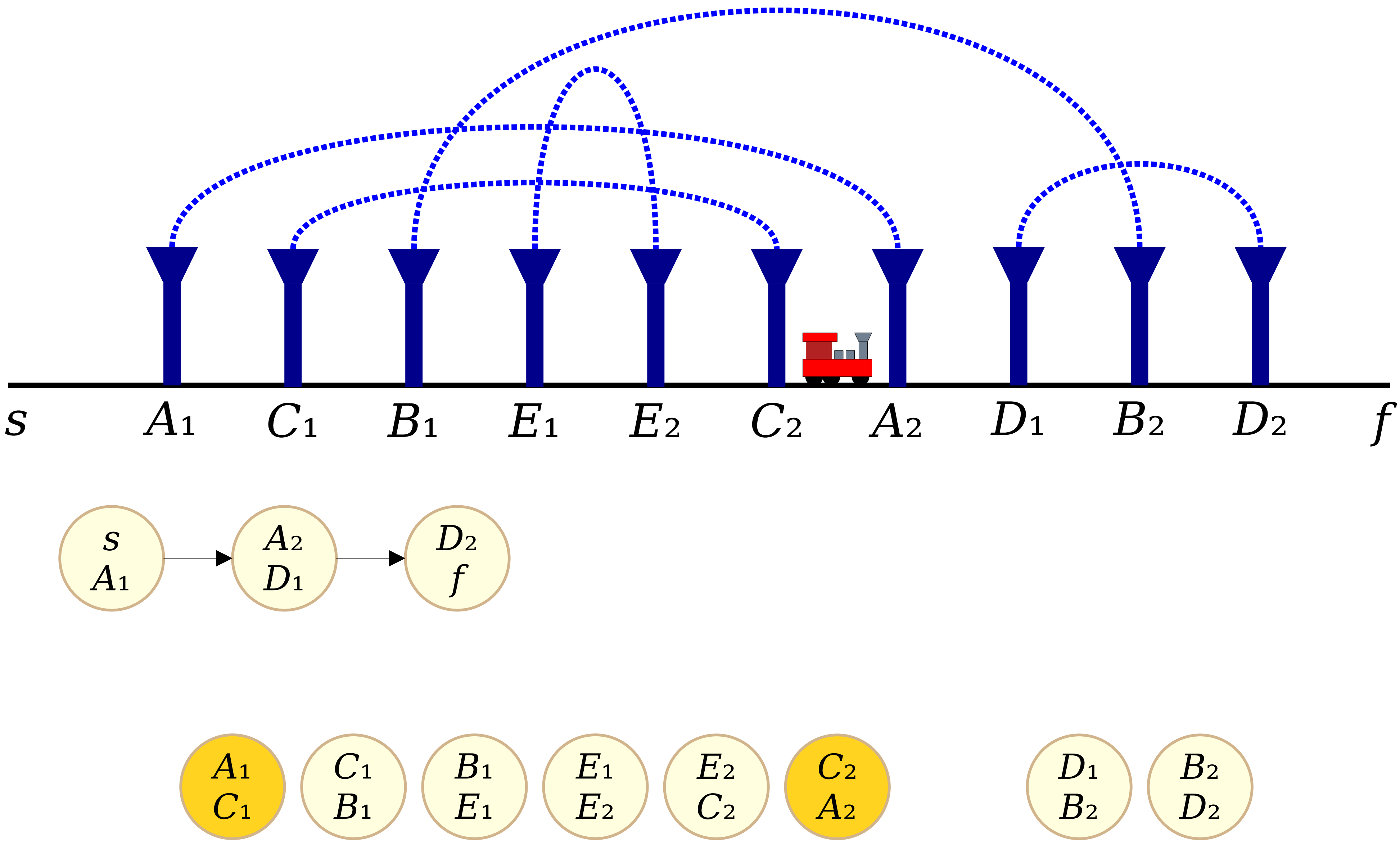
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
D1 B2
B2 D2



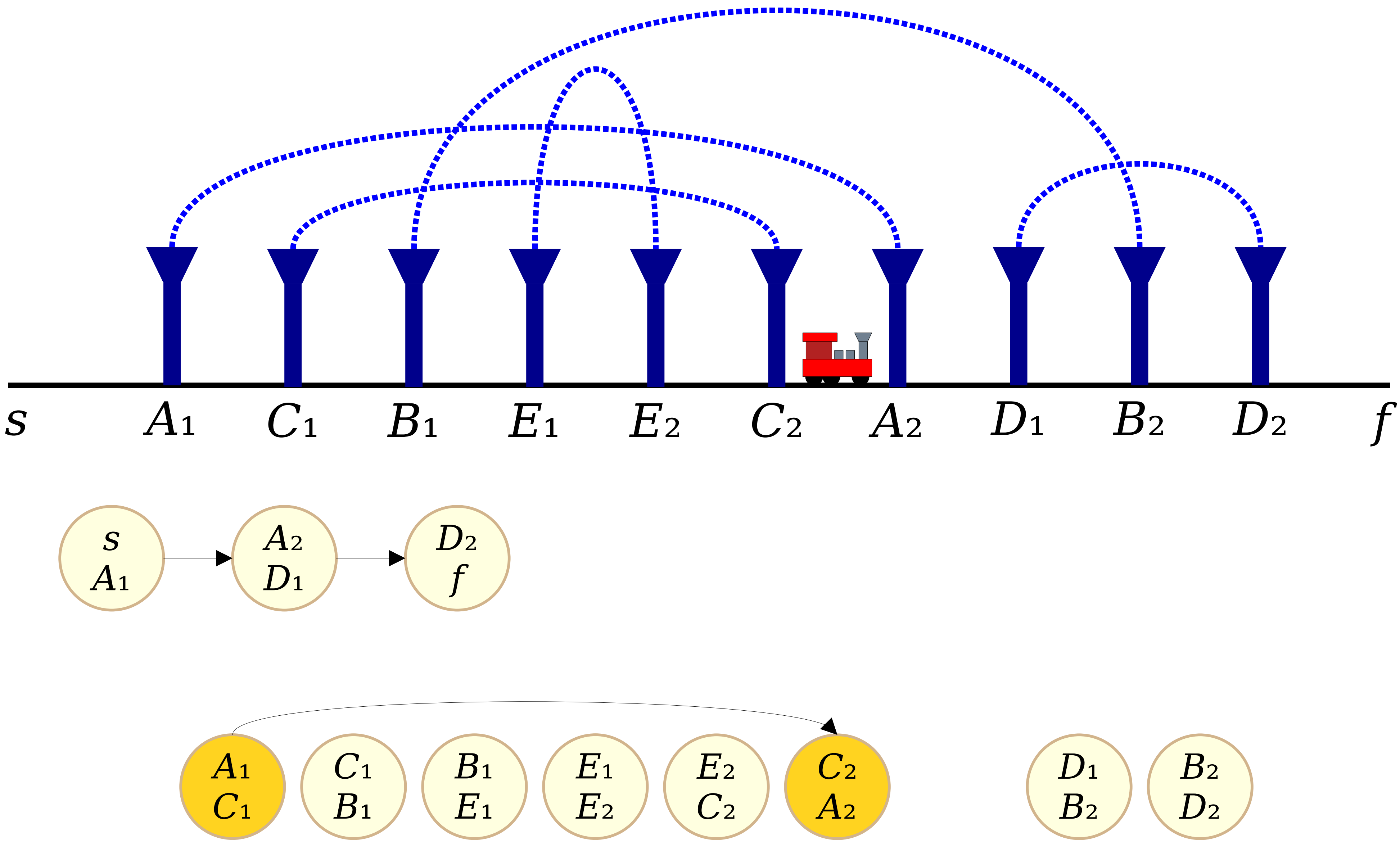
s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
D_1 B_2
B_2 D_2

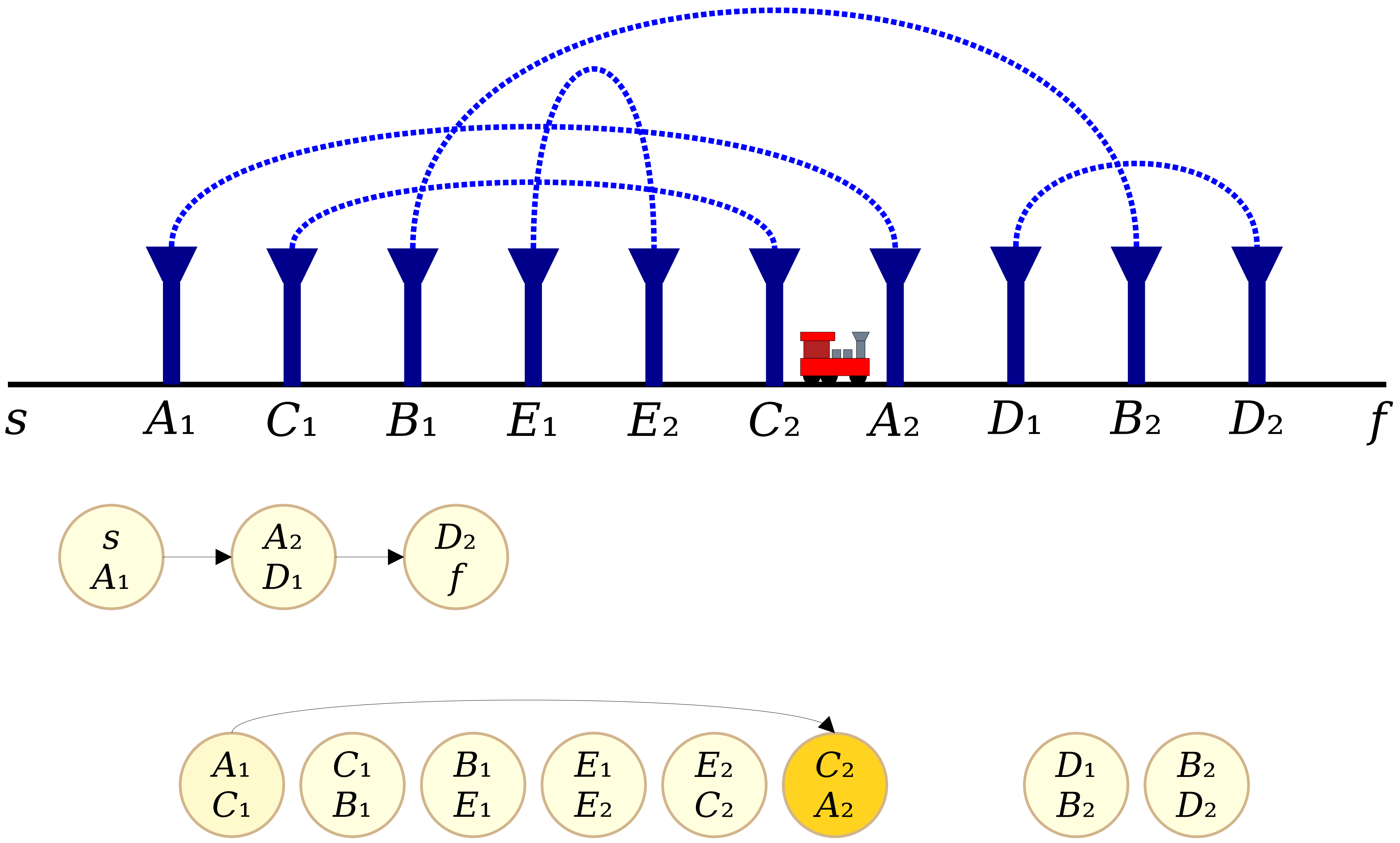


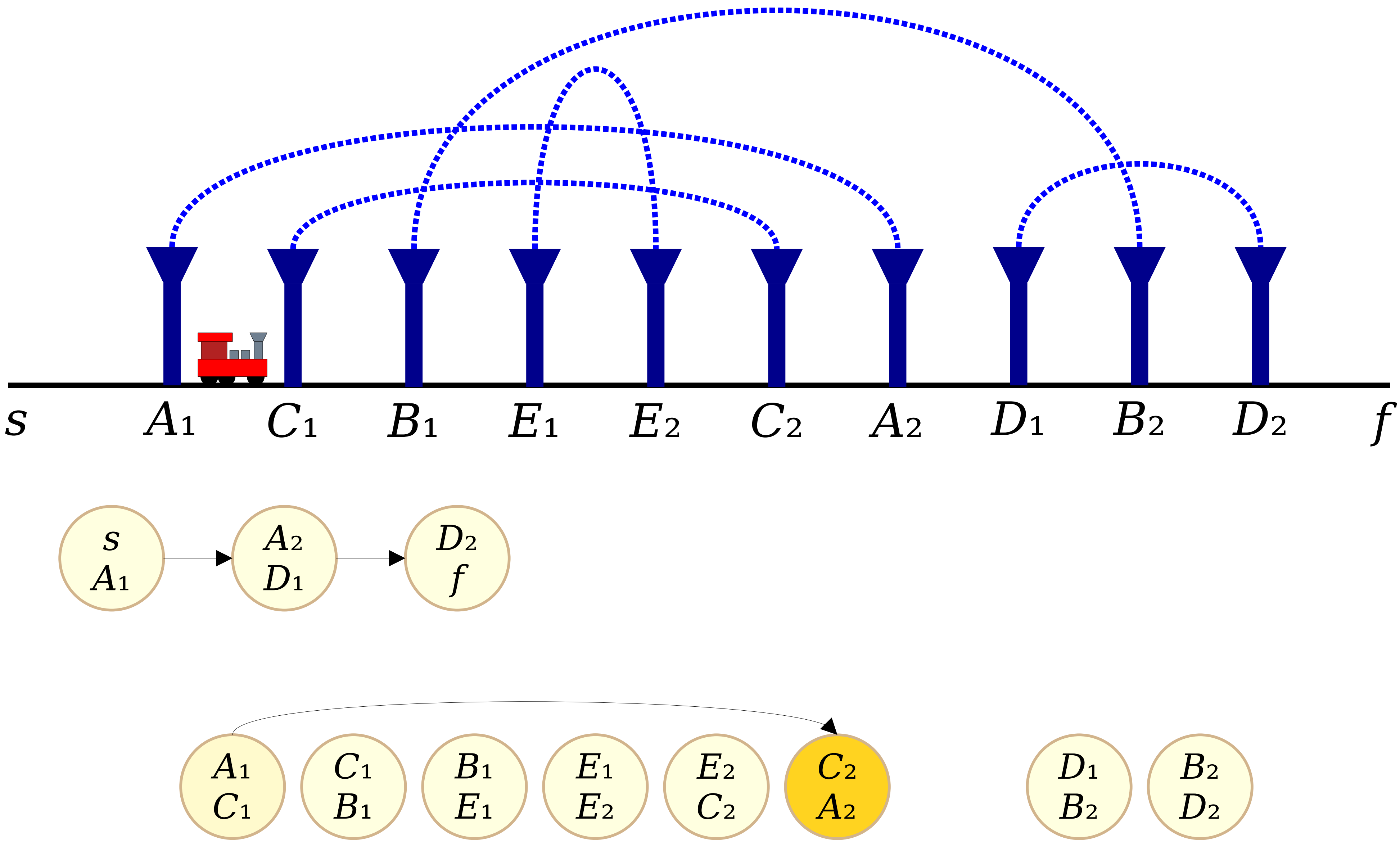
s A_1 C_1 B_1 E_1 E_2 C_2 A_2 D_1 B_2 D_2 f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
D_1 B_2
B_2 D_2

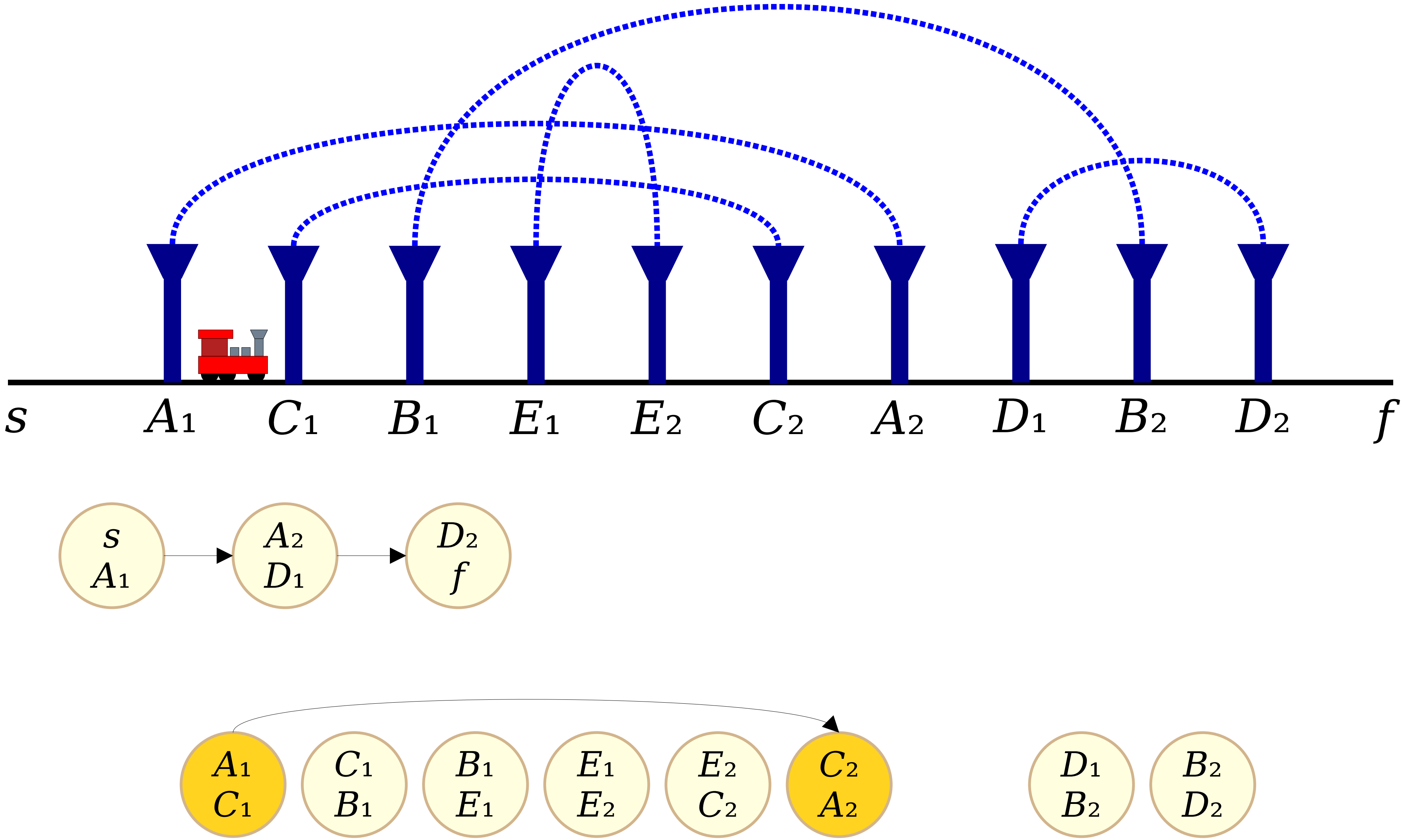


s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
D_1 B_2
B_2 D_2

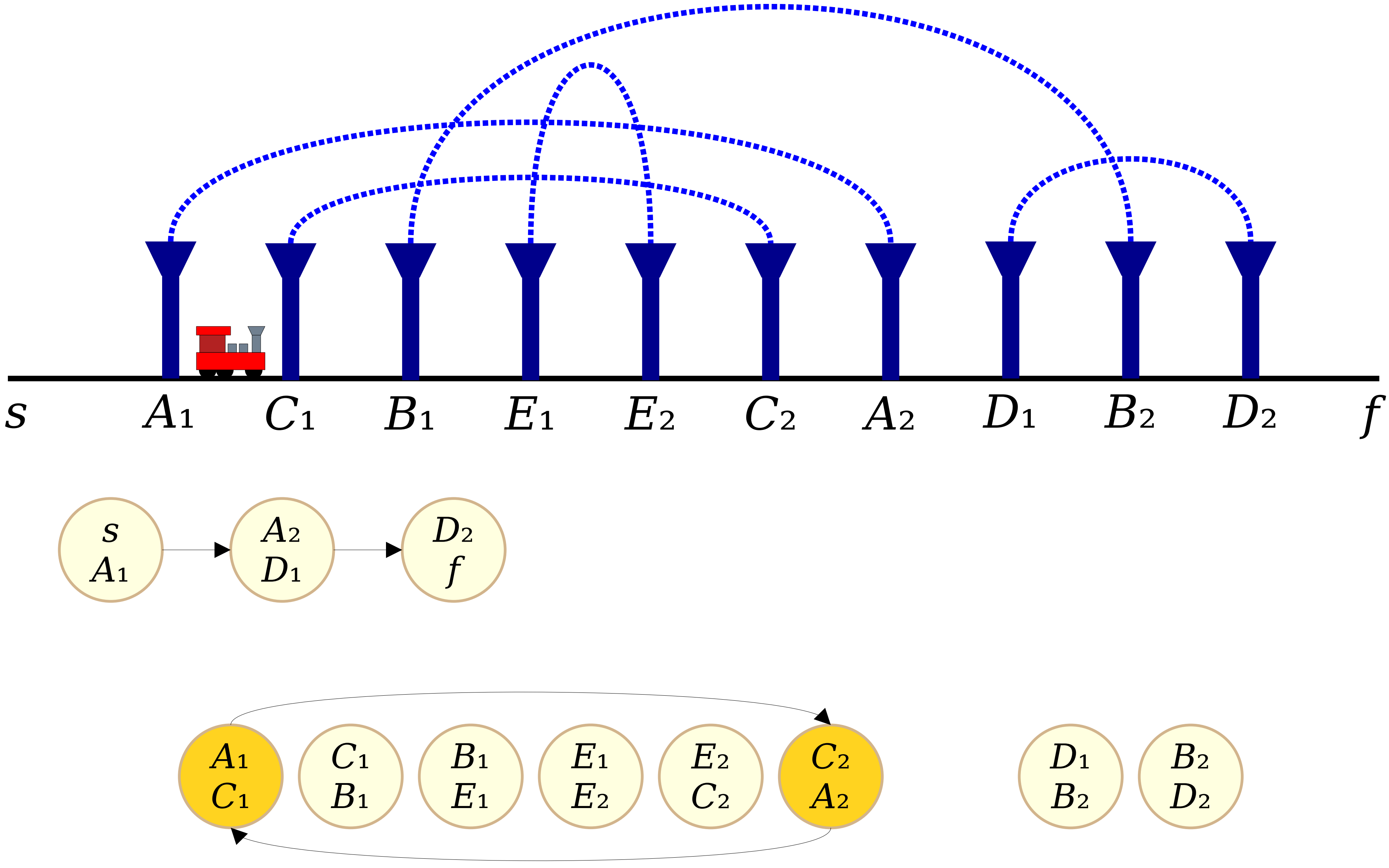

s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
D1 B2
B2 D2


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
D1 B2
B2 D2

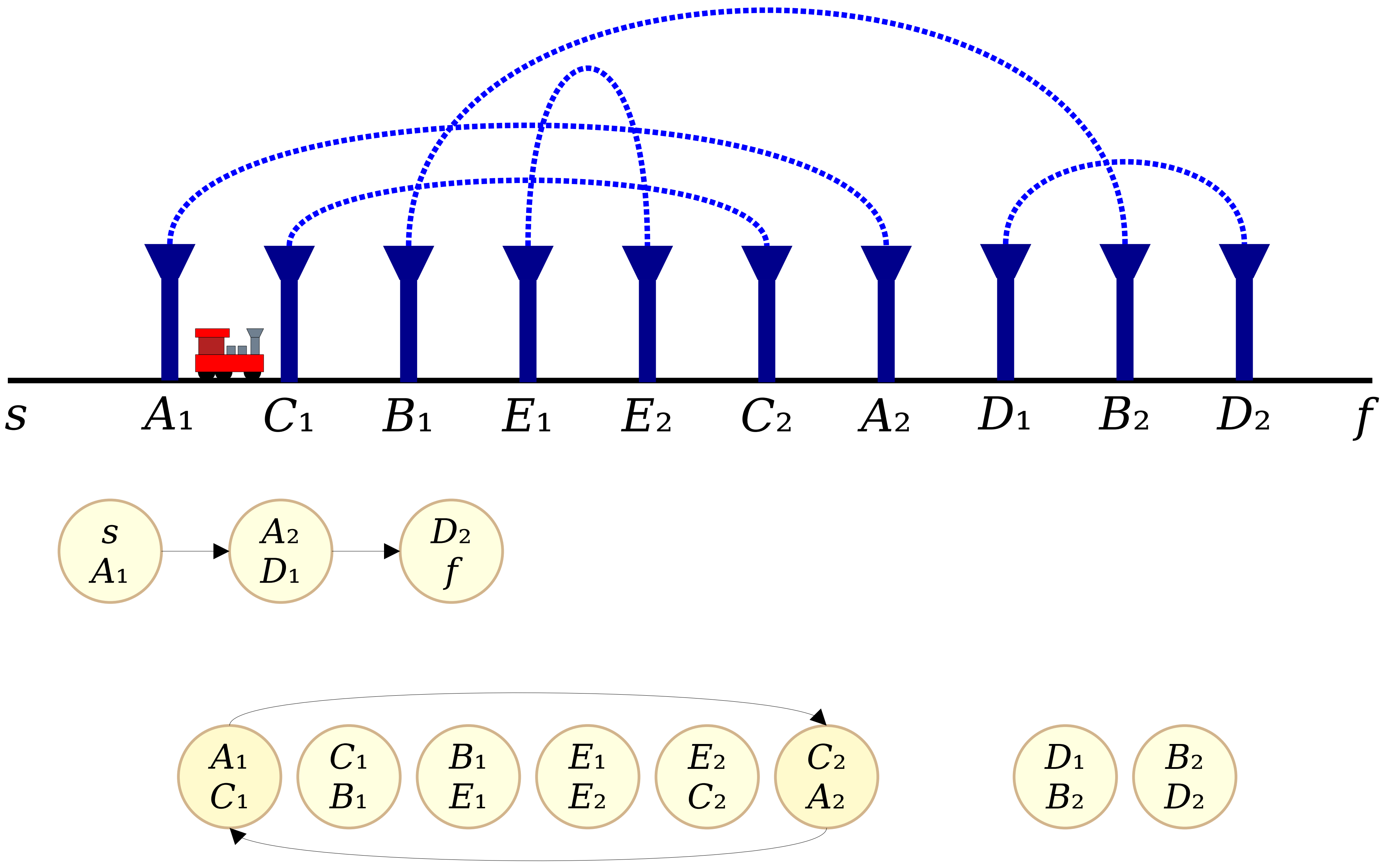

s A_1 C_1 B_1 E_1 E_2 C_2 A_2 D_1 B_2 D_2 f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
D_1 B_2
B_2 D_2



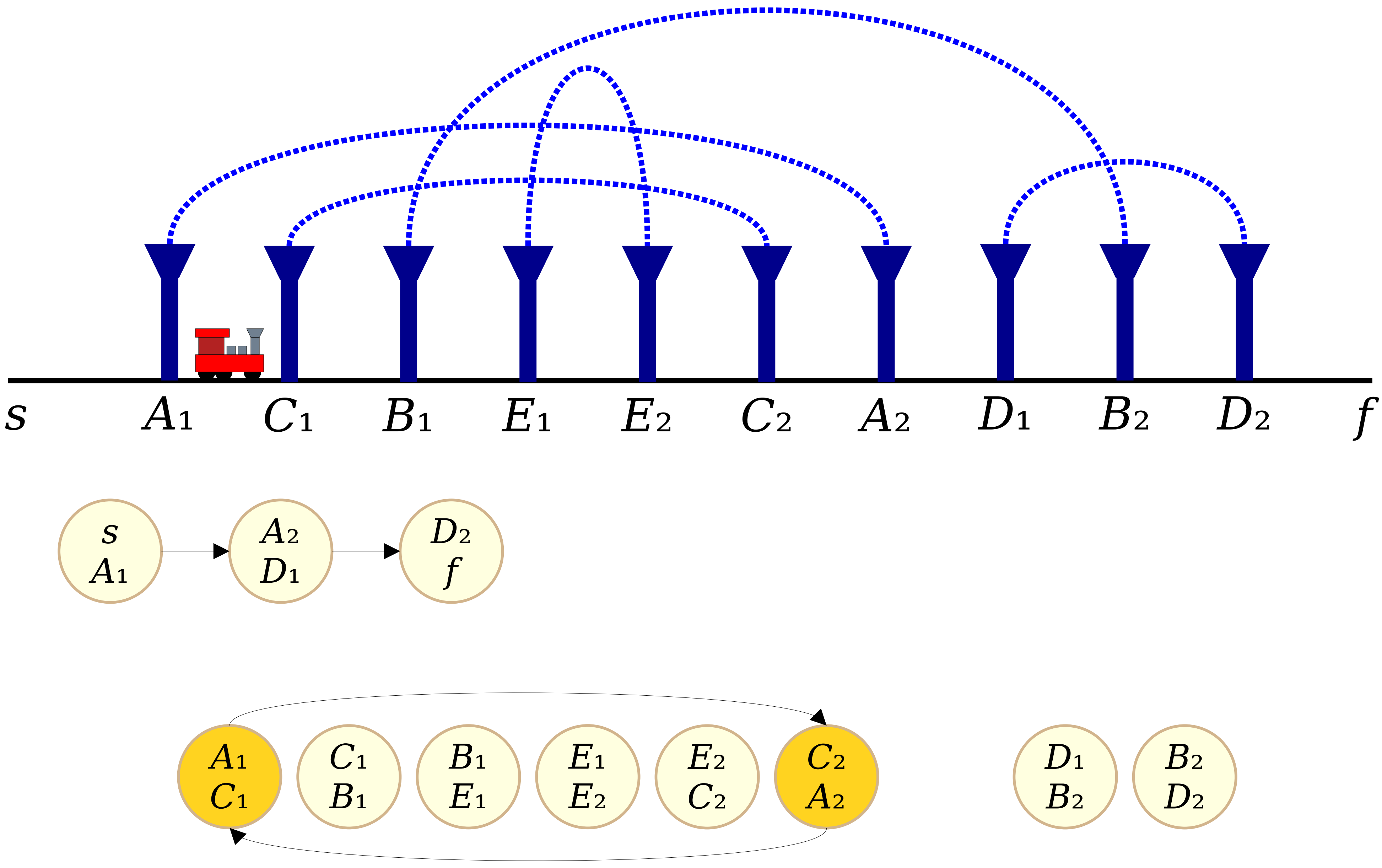
s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
D_1 B_2
B_2 D_2

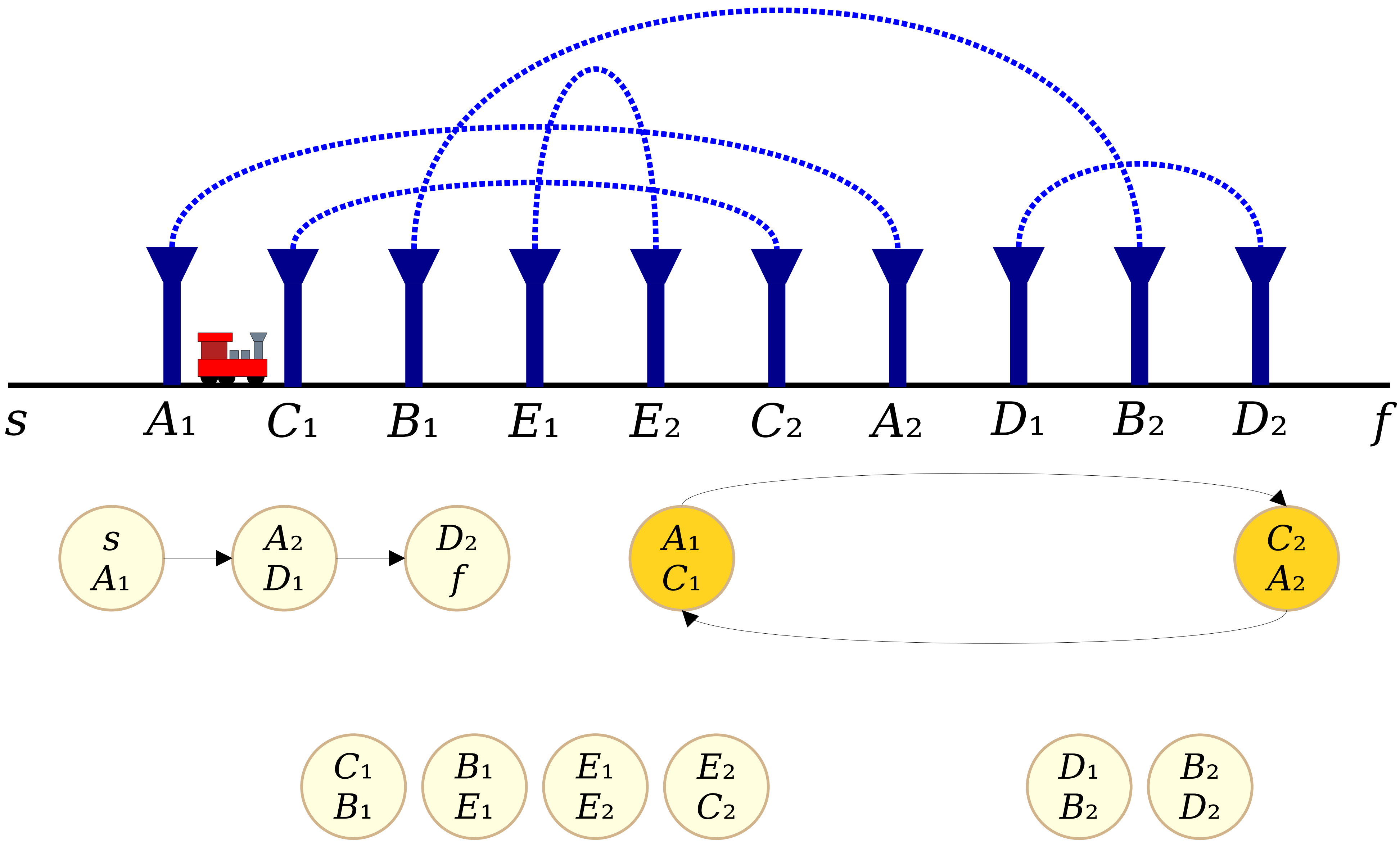


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C1 B1
B1 E1
E1 E2
E2 C2
C2 A2
D1 B2
B2 D2

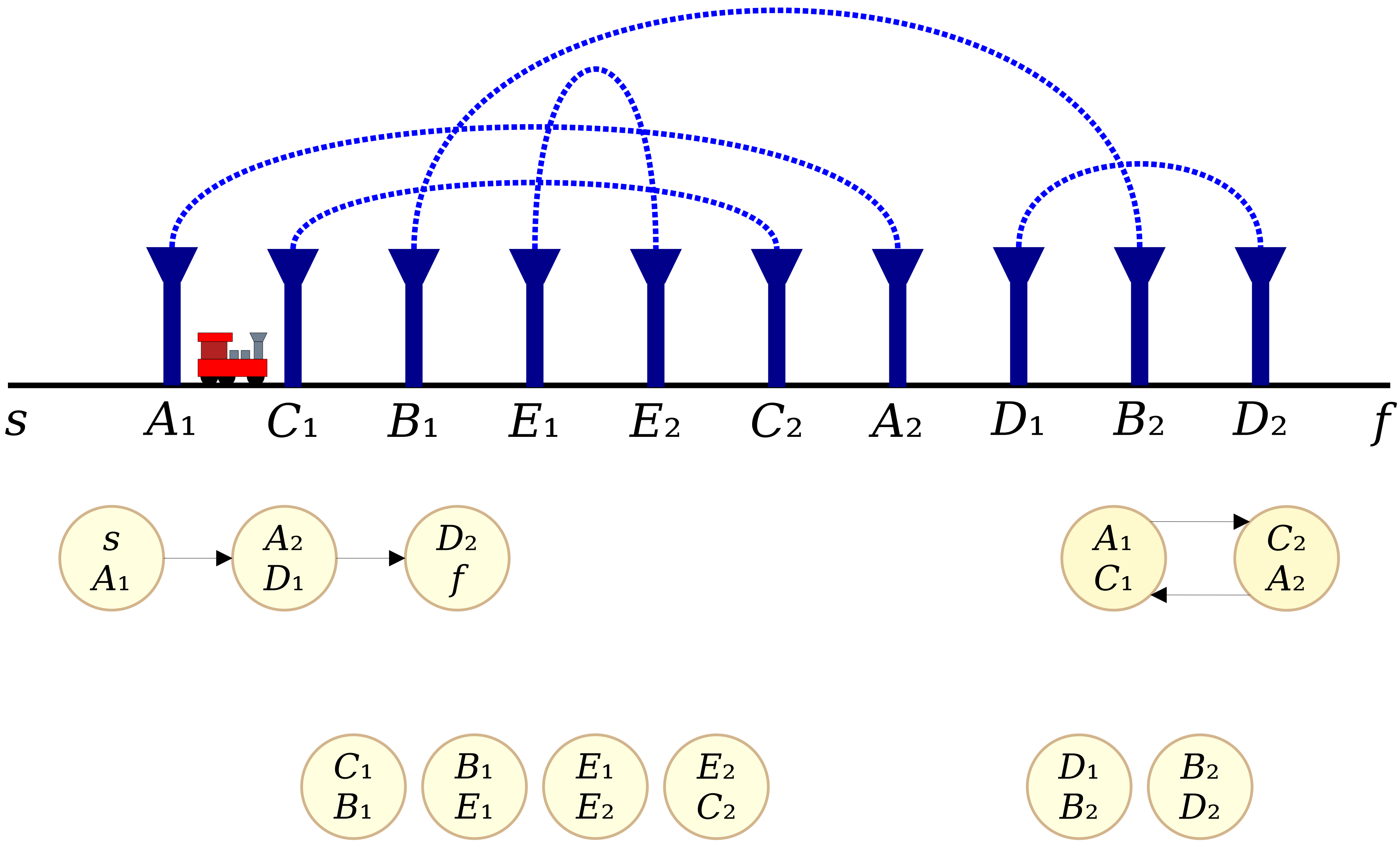


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f

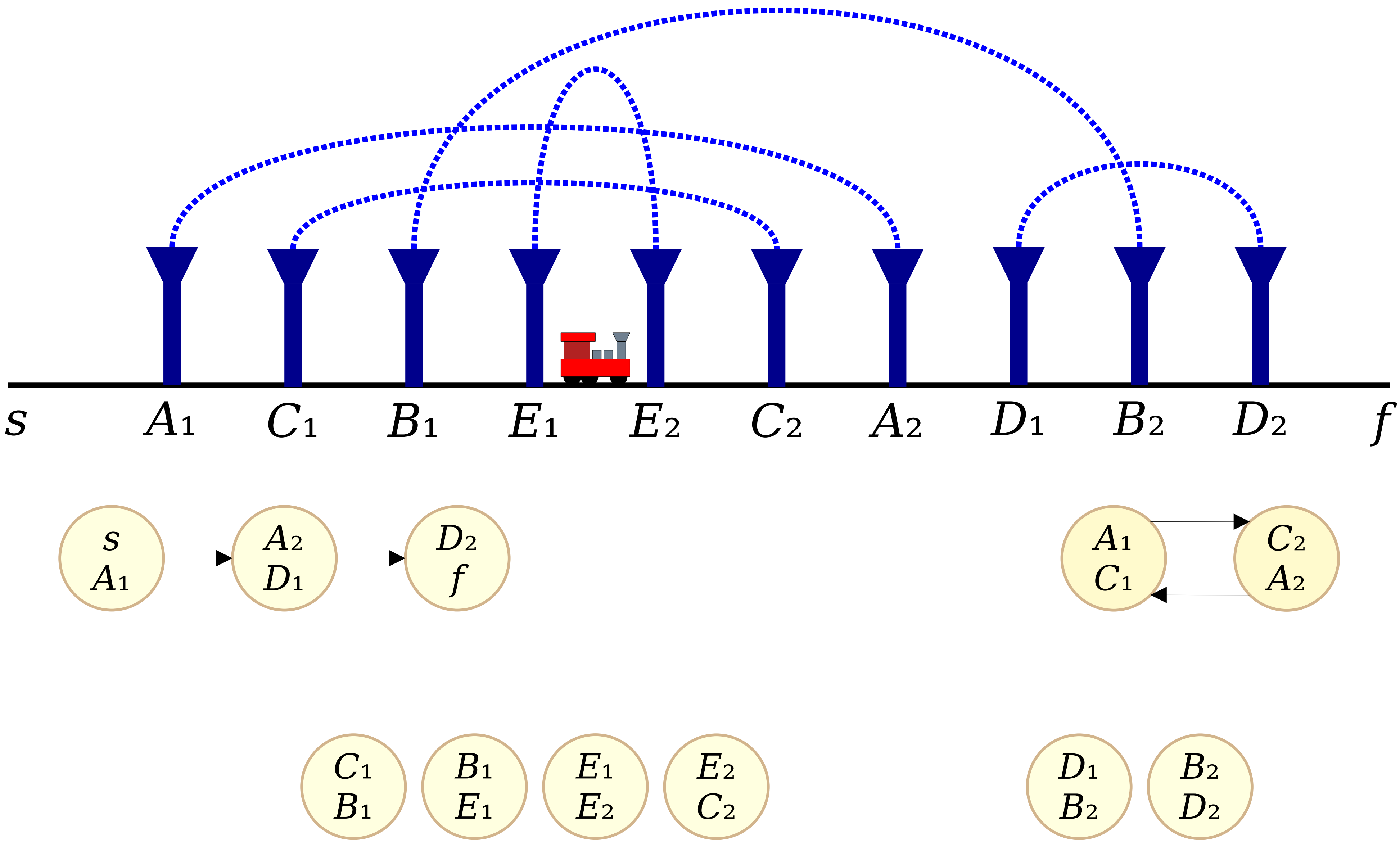

s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
C_2 A_2
D_1 B_2
B_2 D_2



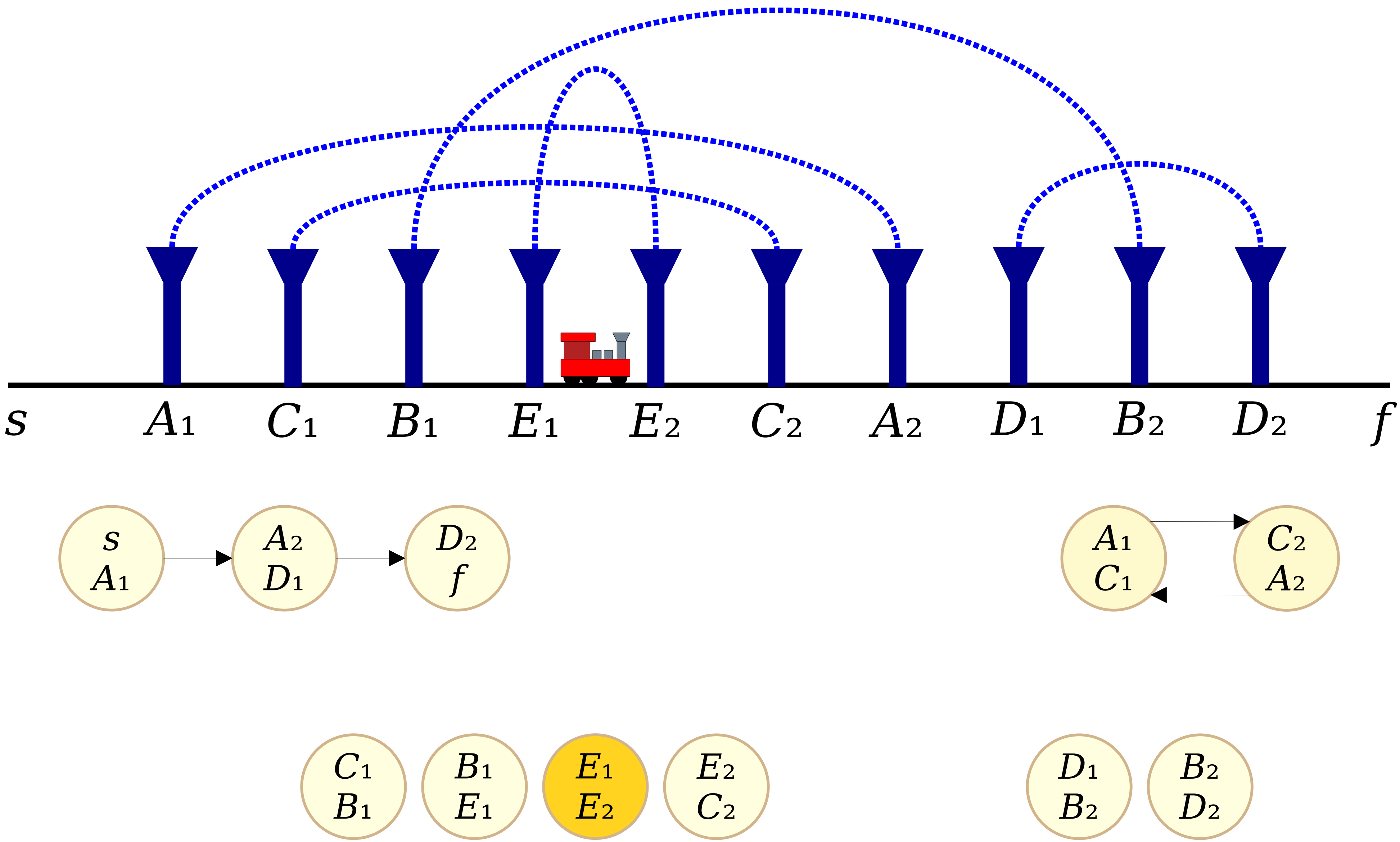
s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_2 A_2
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
D_1 B_2
B_2 D_2



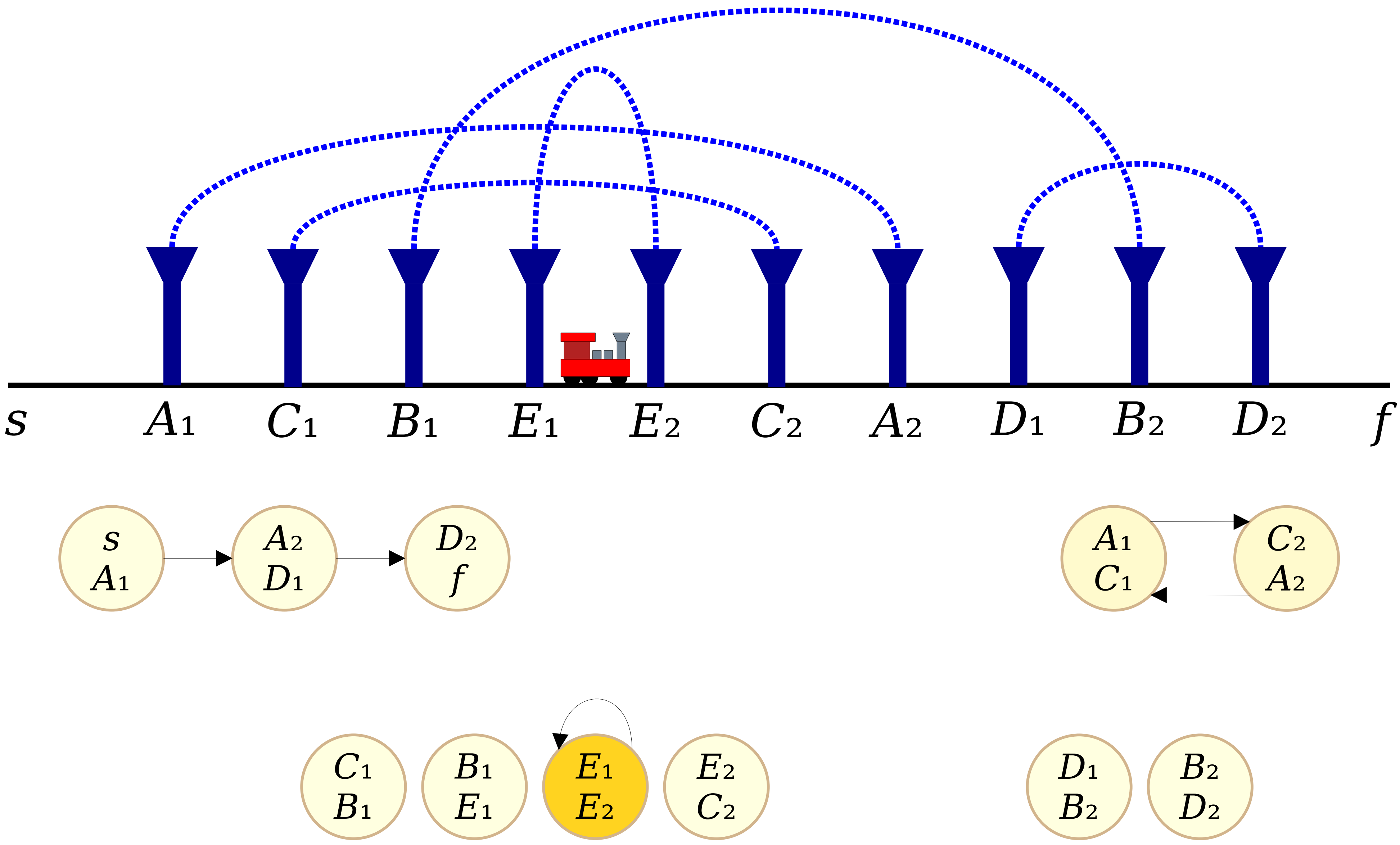
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E1 E2
E2 C2
D1 B2
B2 D2

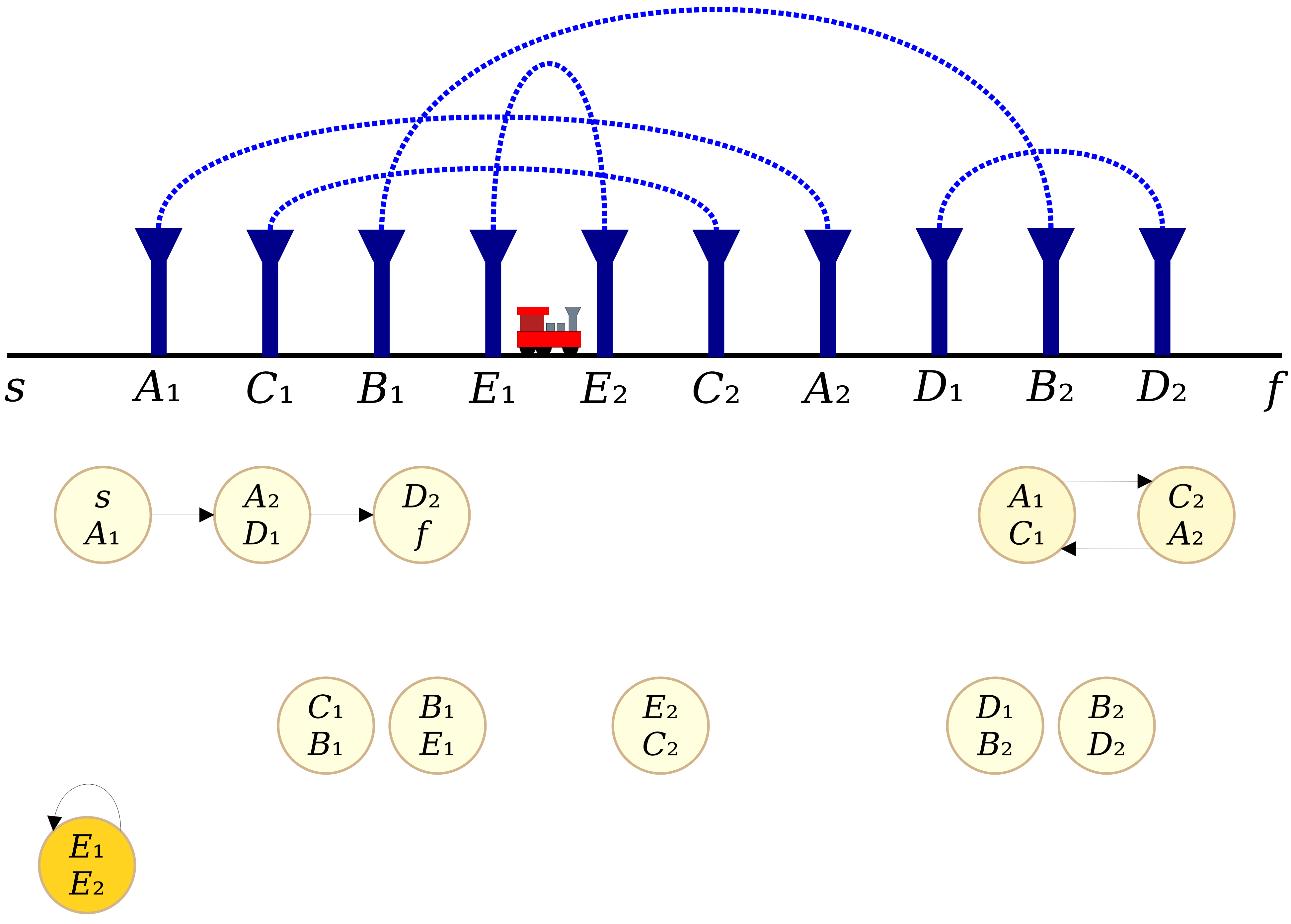


s A_1 C_1 B_1 E_1 E_2 C_2 A_2 D_1 B_2 D_2 f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_2 A_2
C_1 B_1
B_1 E_1
E_1 E_2
E_2 C_2
D_1 B_2
B_2 D_2

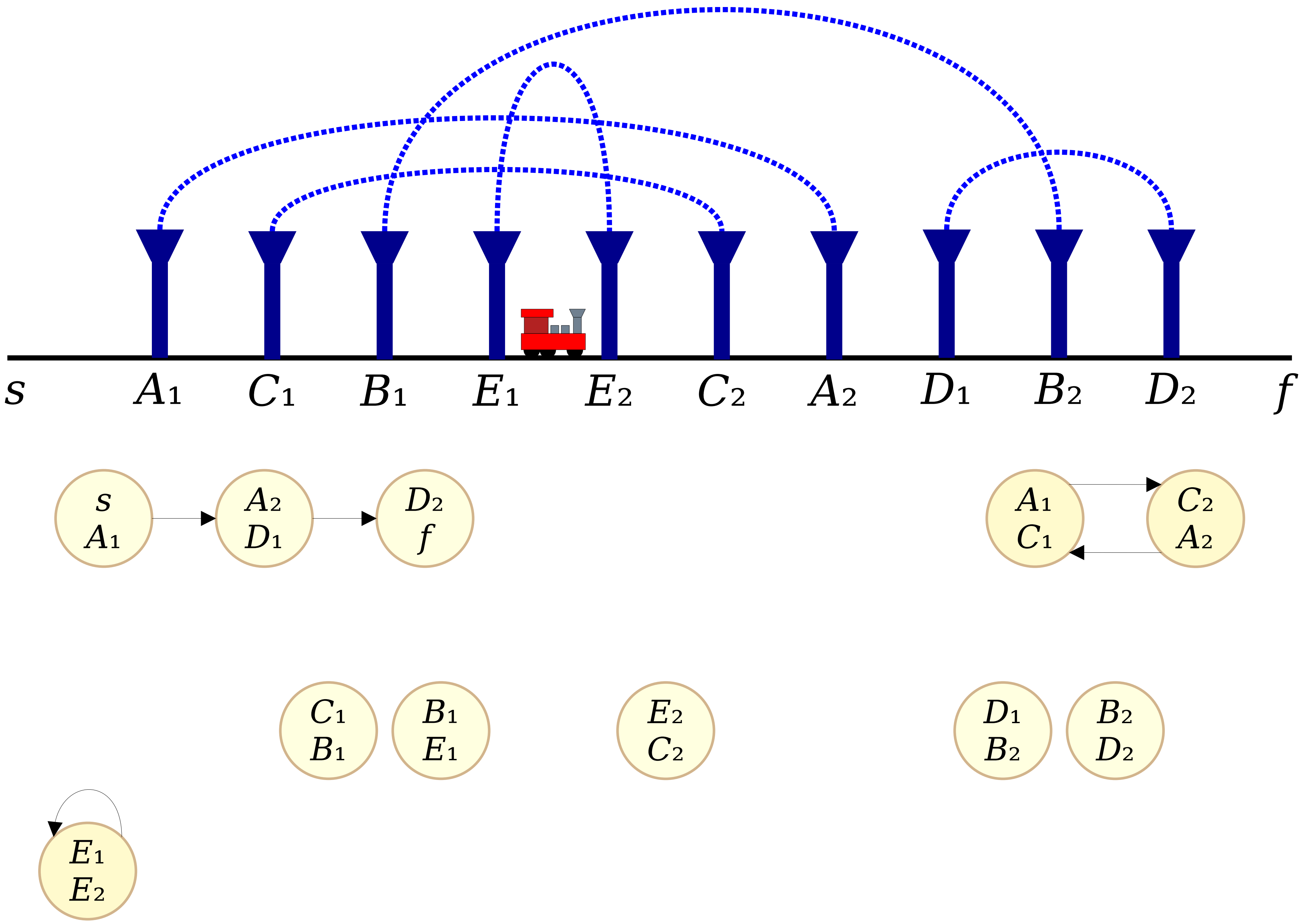


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E1 E2
E2 C2
D1 B2
B2 D2

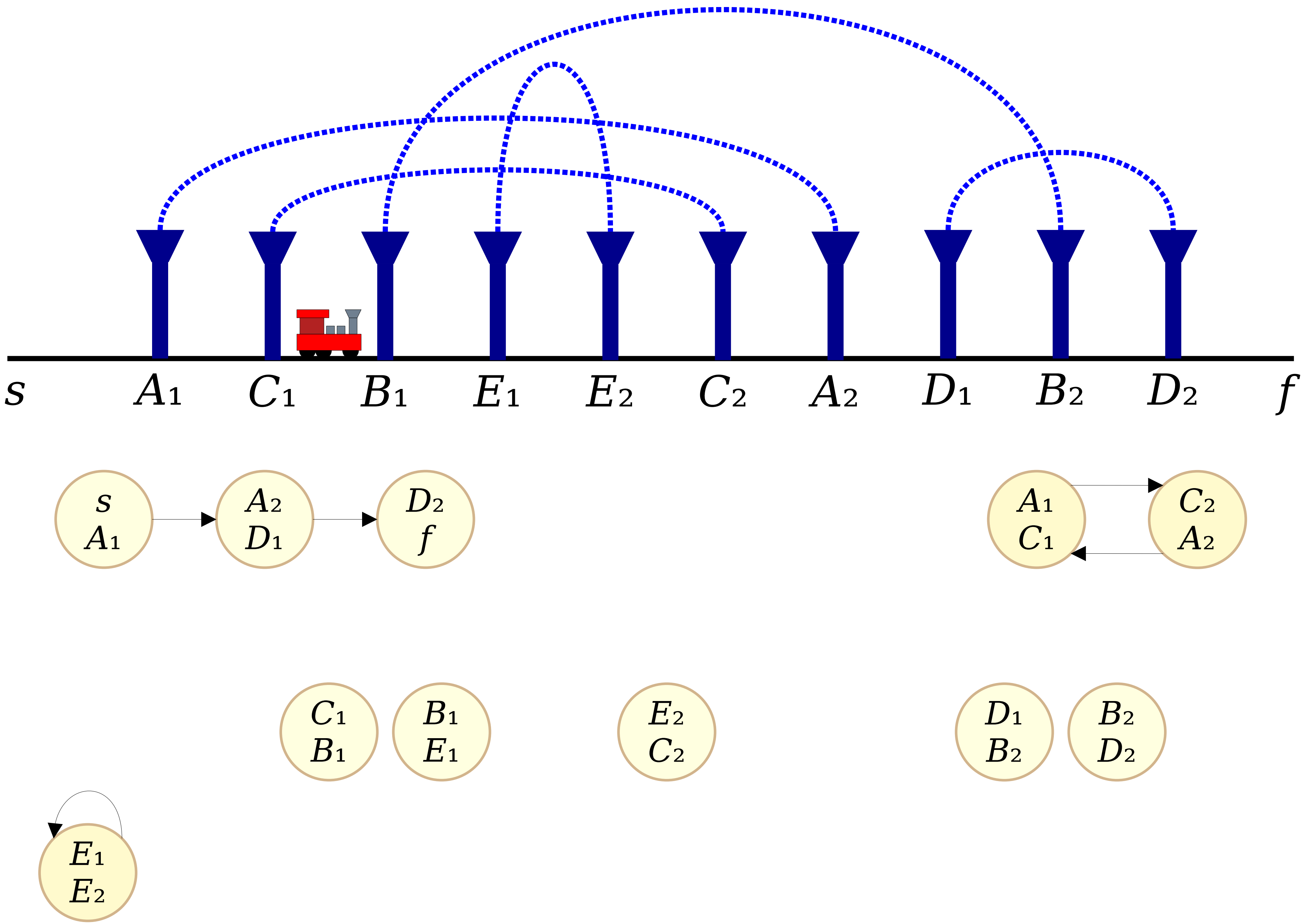

s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E1 E2
E2 C2
D1 B2
B2 D2



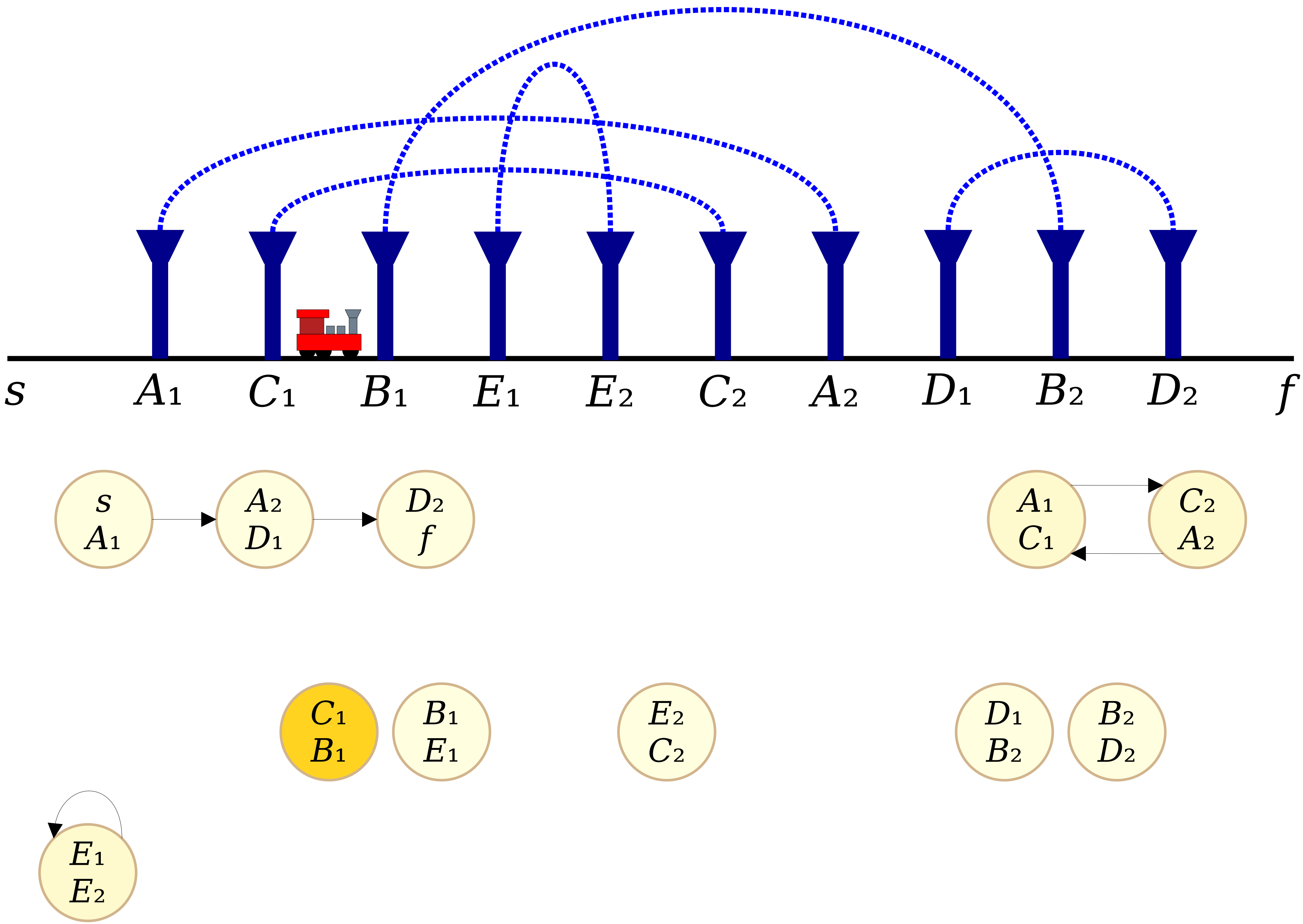
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2

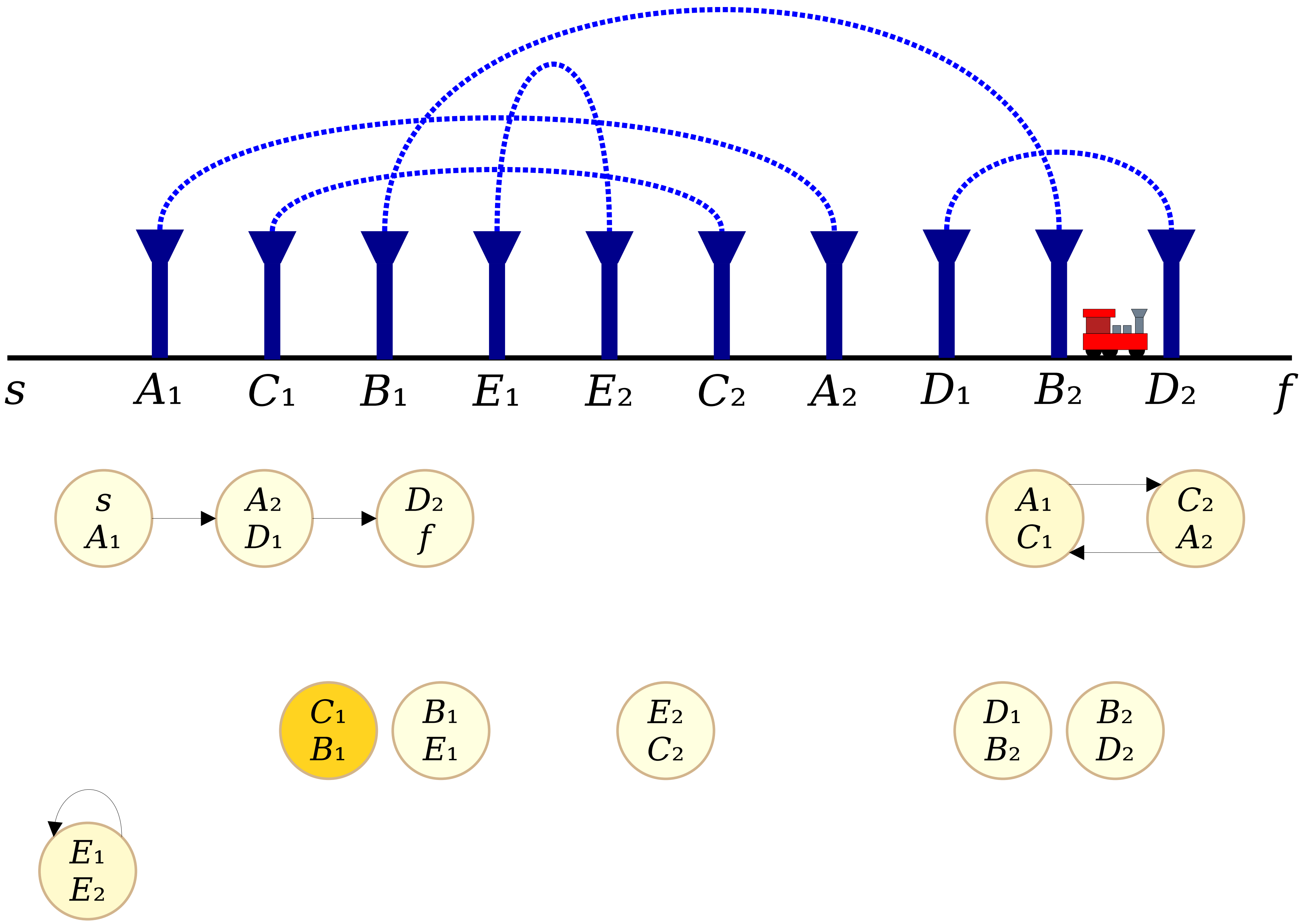


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2

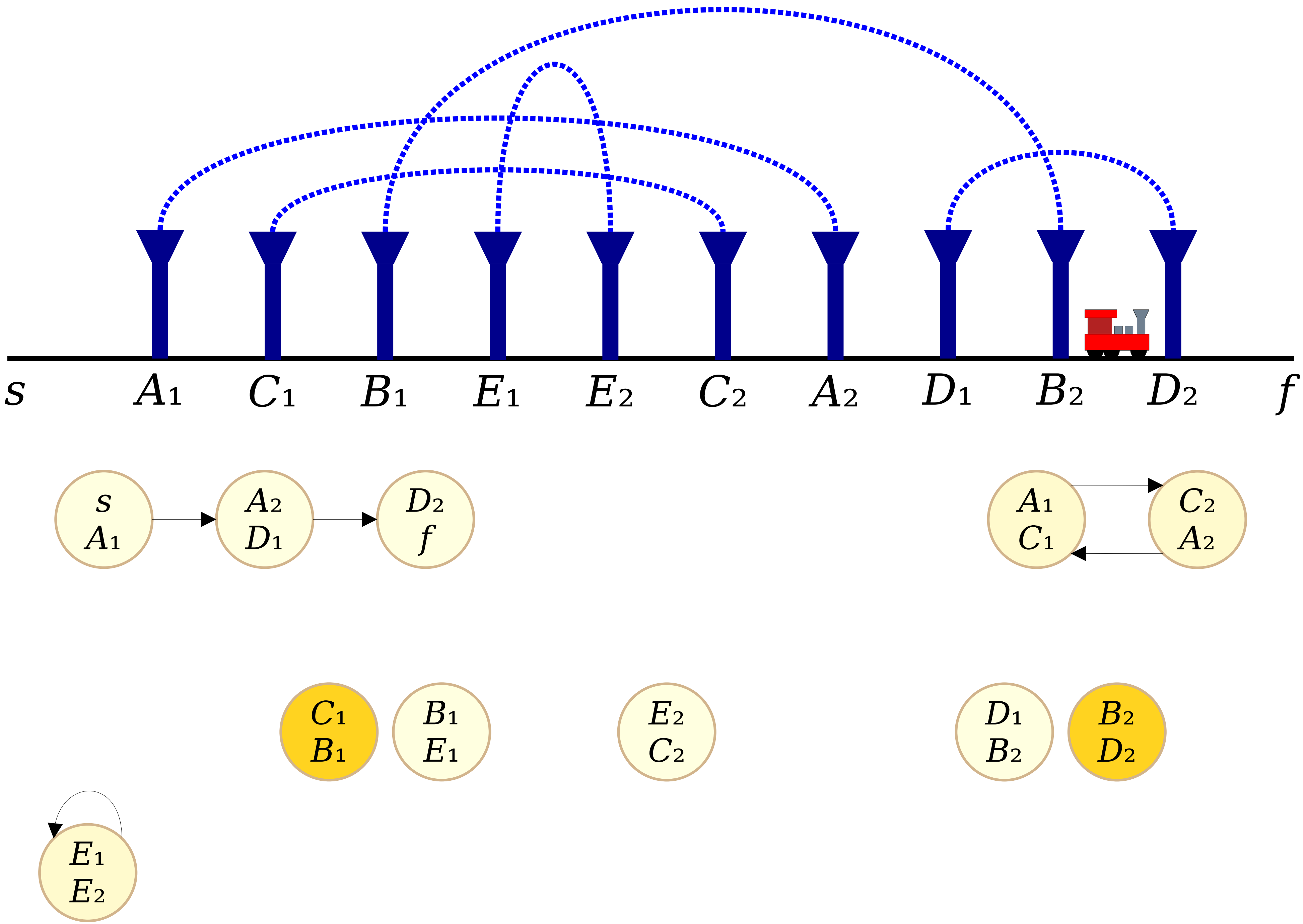


s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_2 A_2
C_1 B_1
B_1 E_1
E_2 C_2
D_1 B_2
B_2 D_2
E_1 E_2

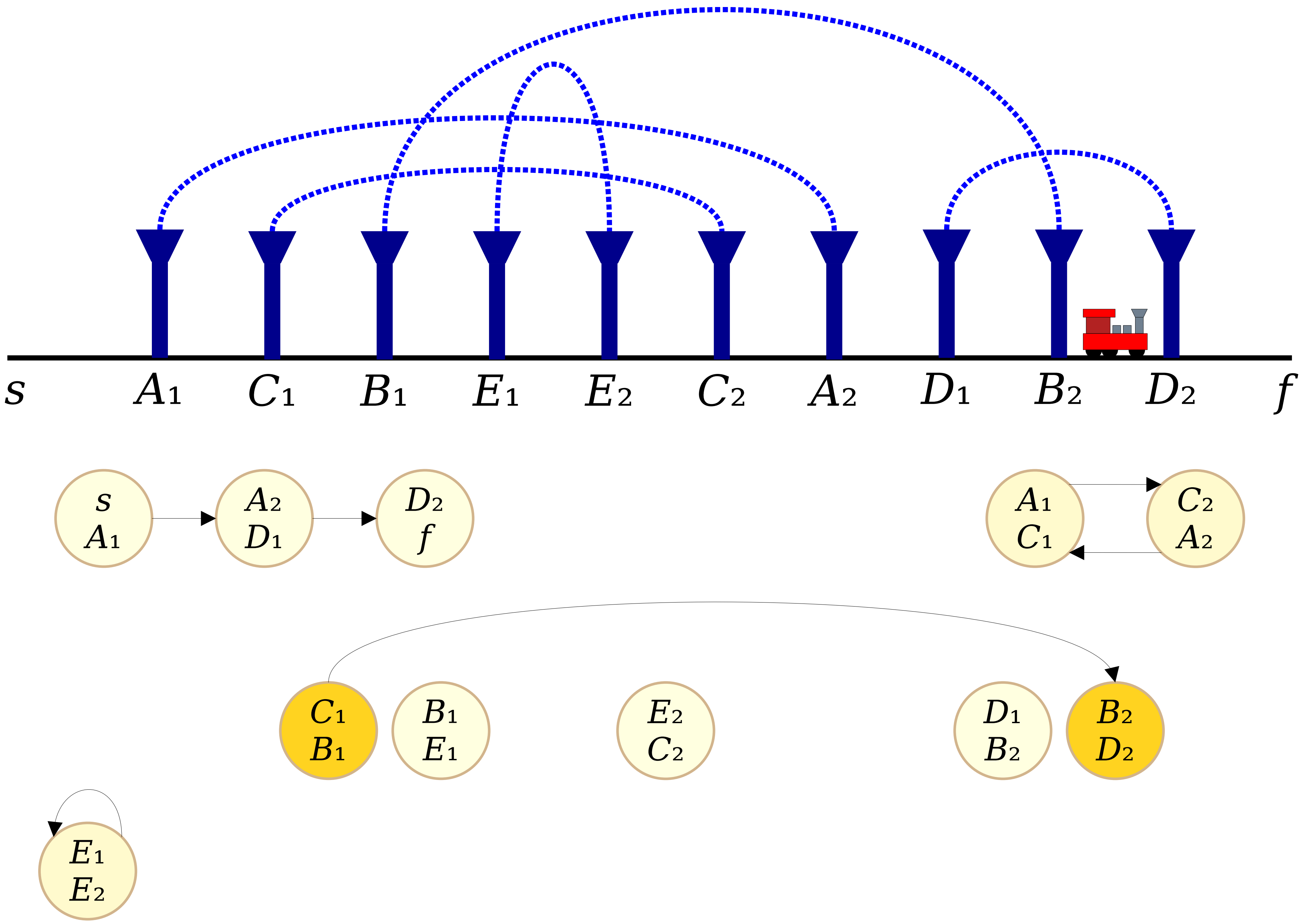

s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2

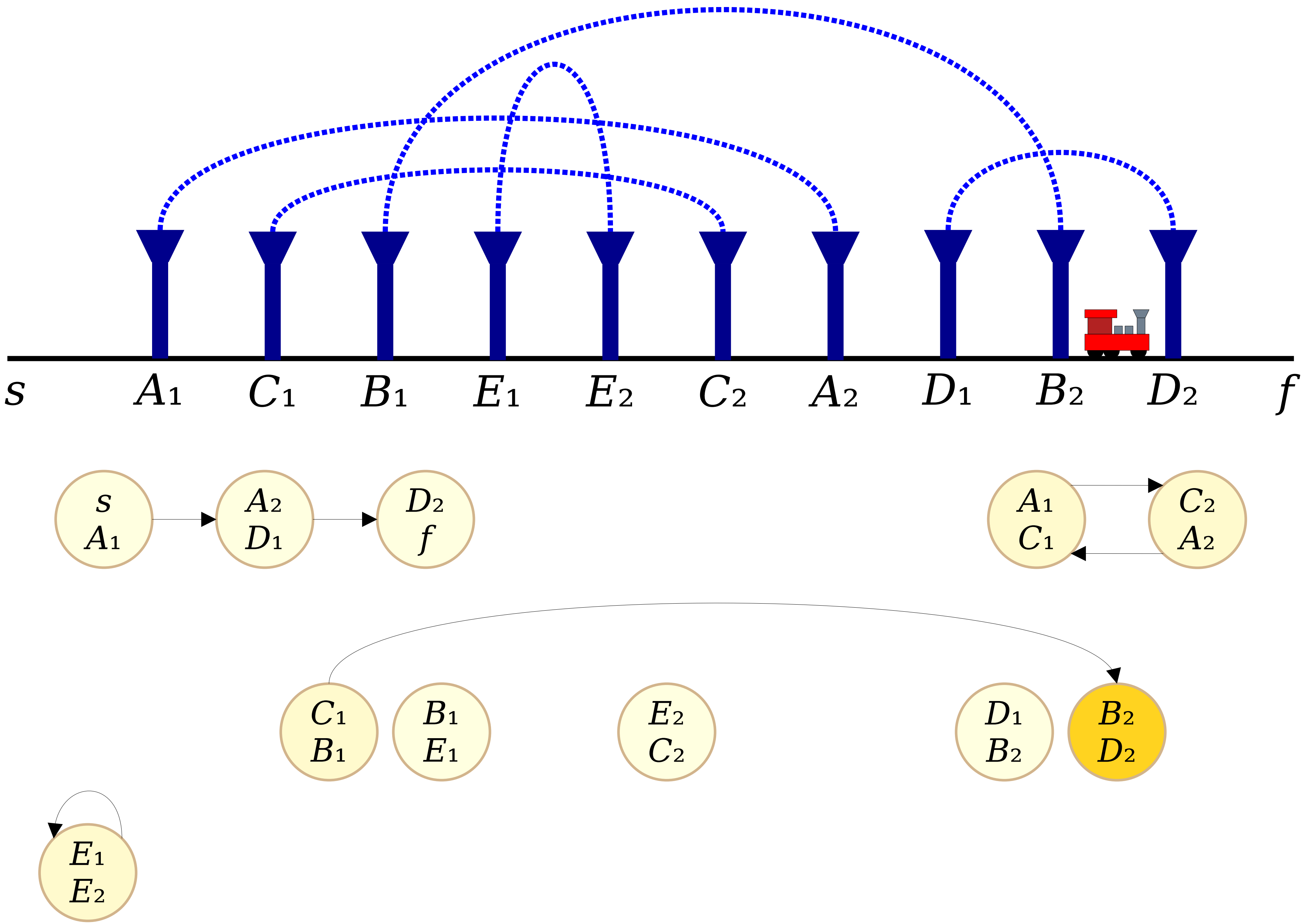


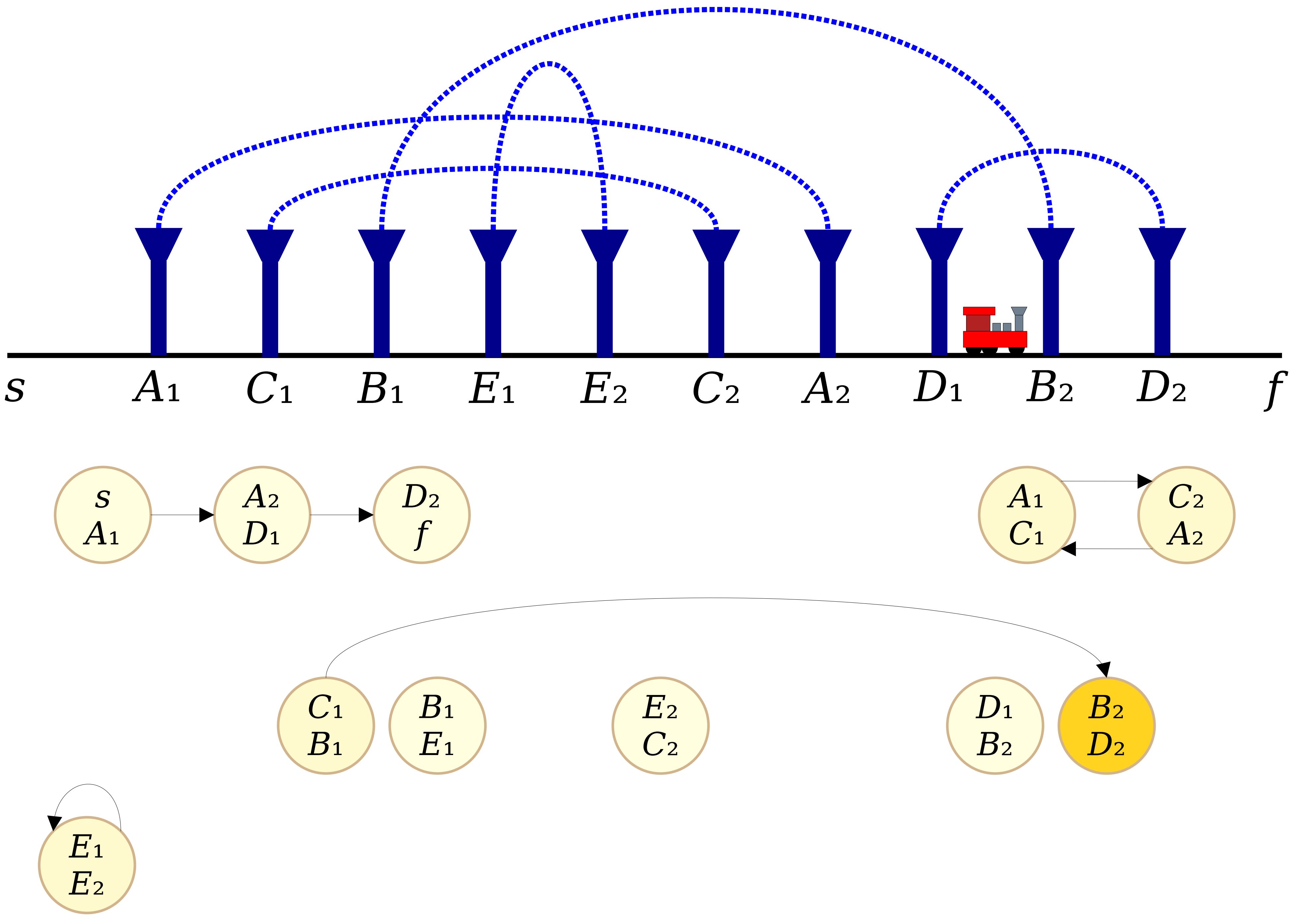
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2



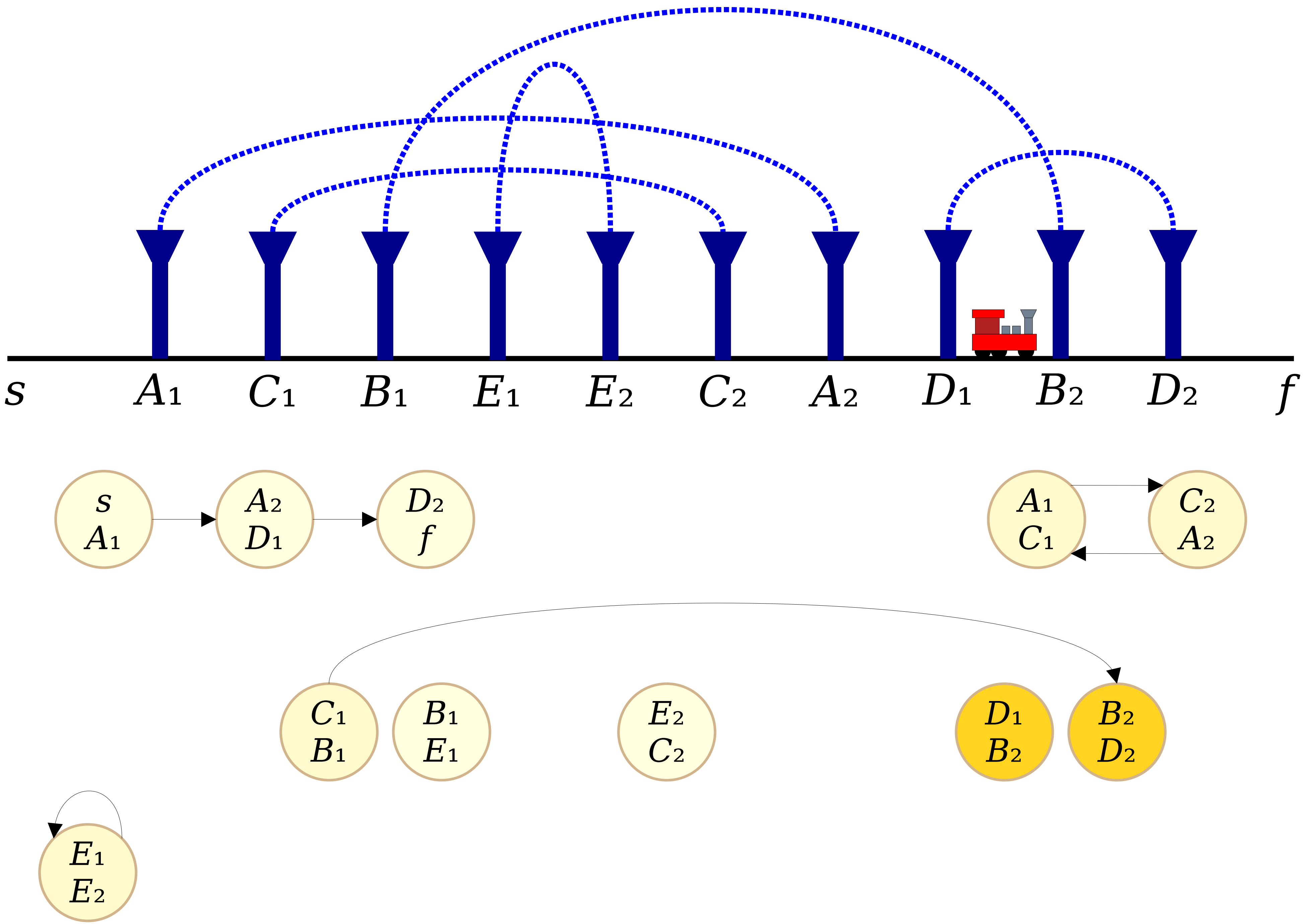
s A_1 C_1 B_1 E_1 E_2 C_2 A_2 D_1 B_2 D_2 f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_2 A_2
C_1 B_1
B_1 E_1
E_2 C_2
D_1 B_2
B_2 D_2
E_1 E_2


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2

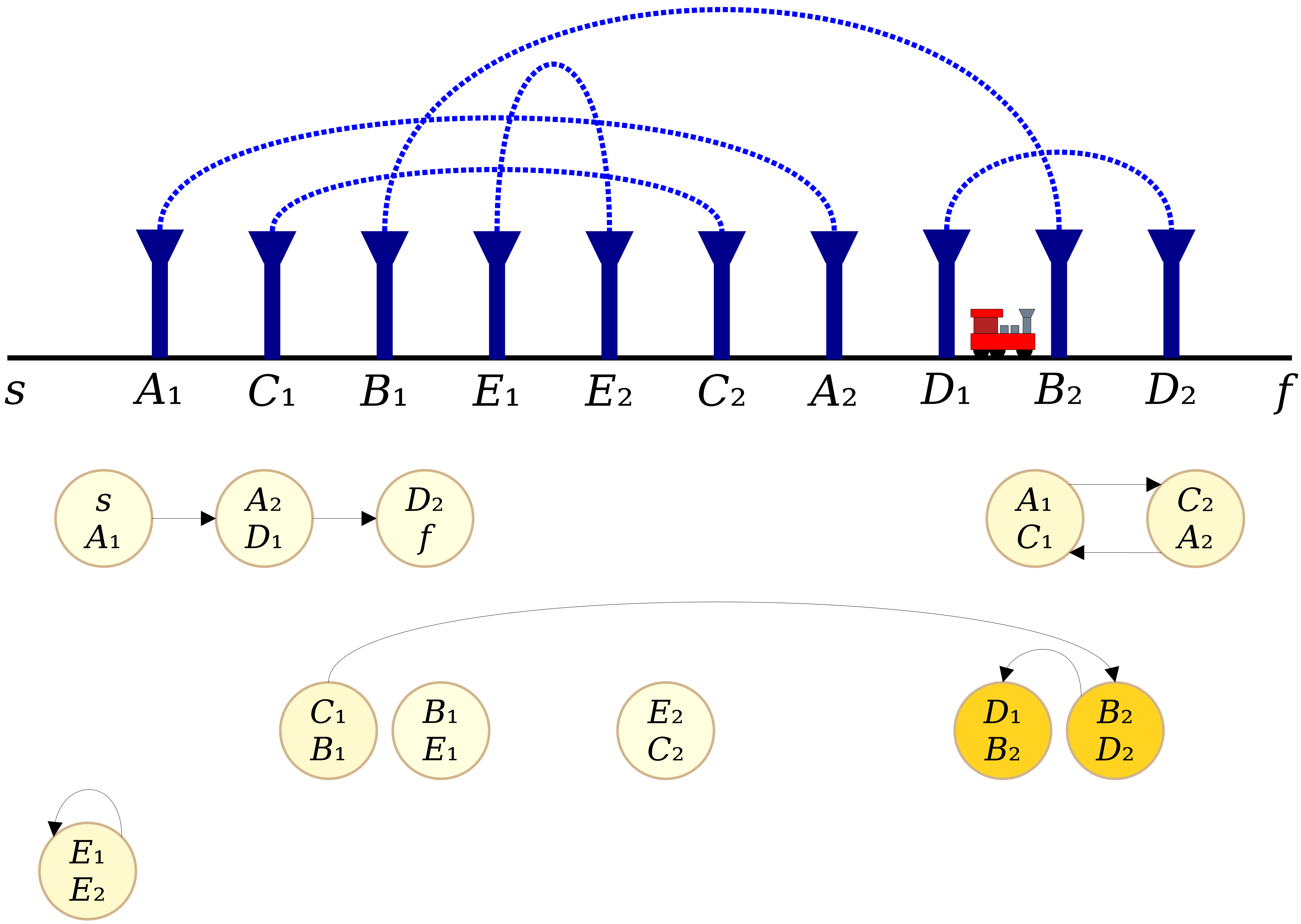

s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2



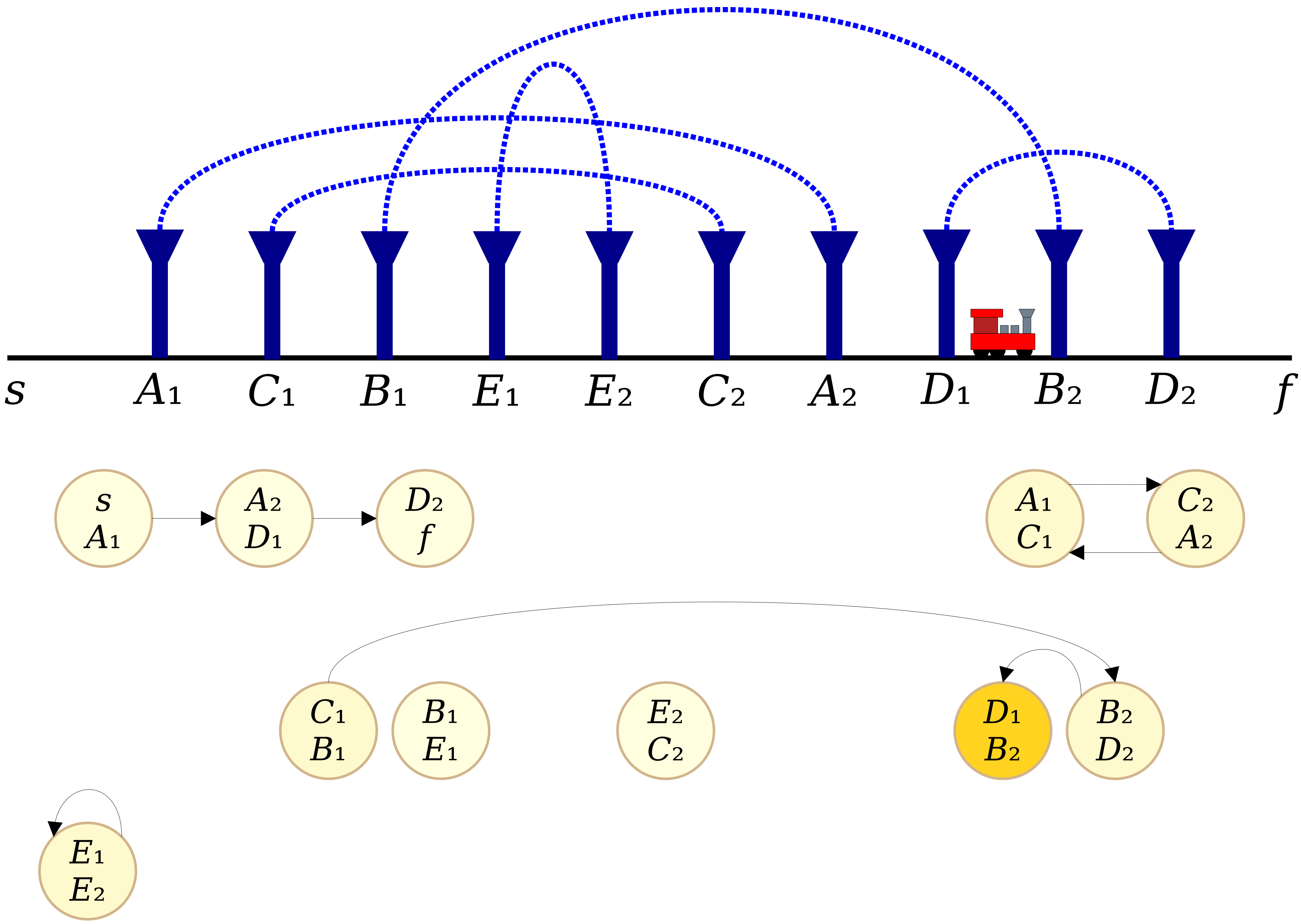
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2



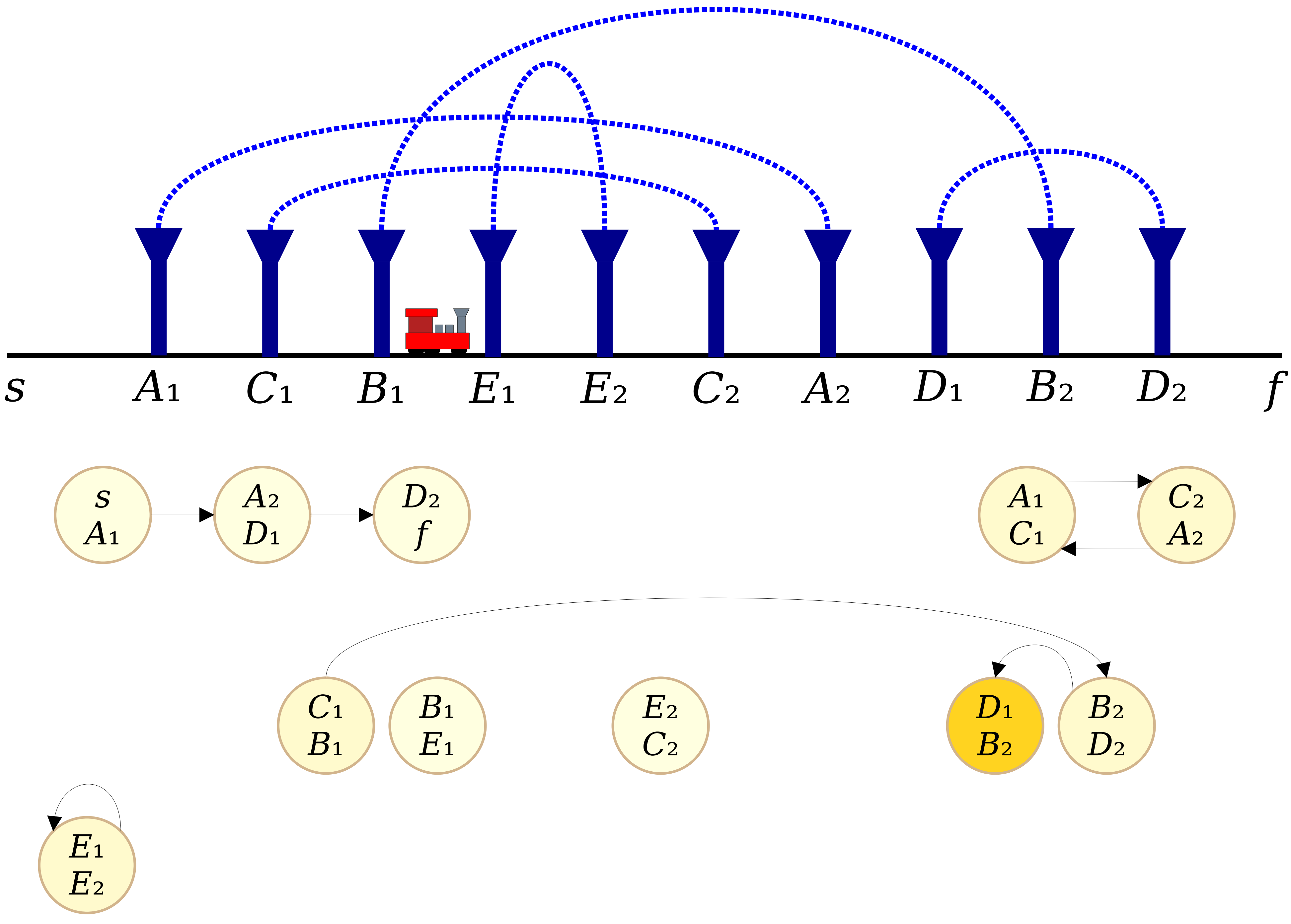
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2



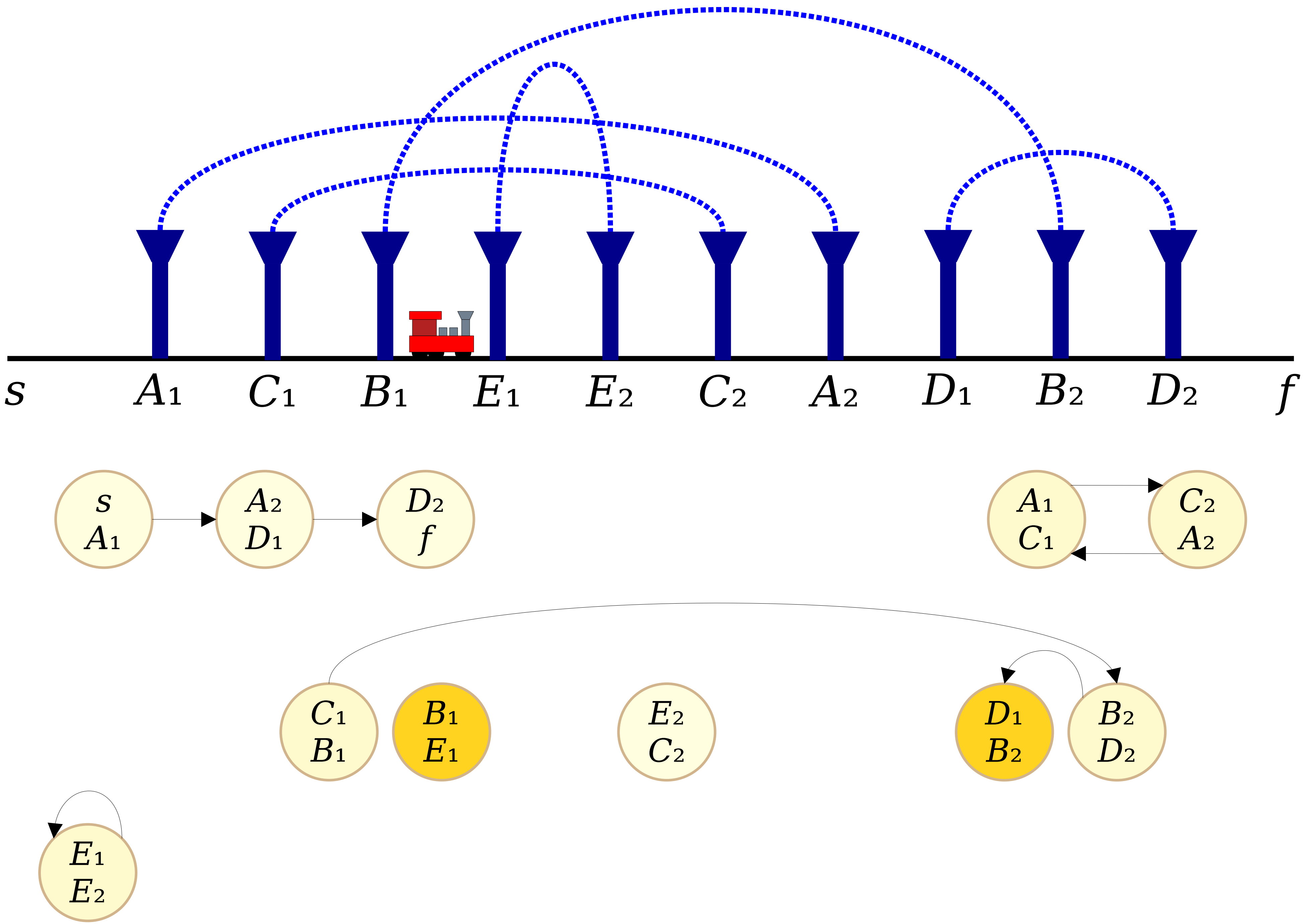
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2



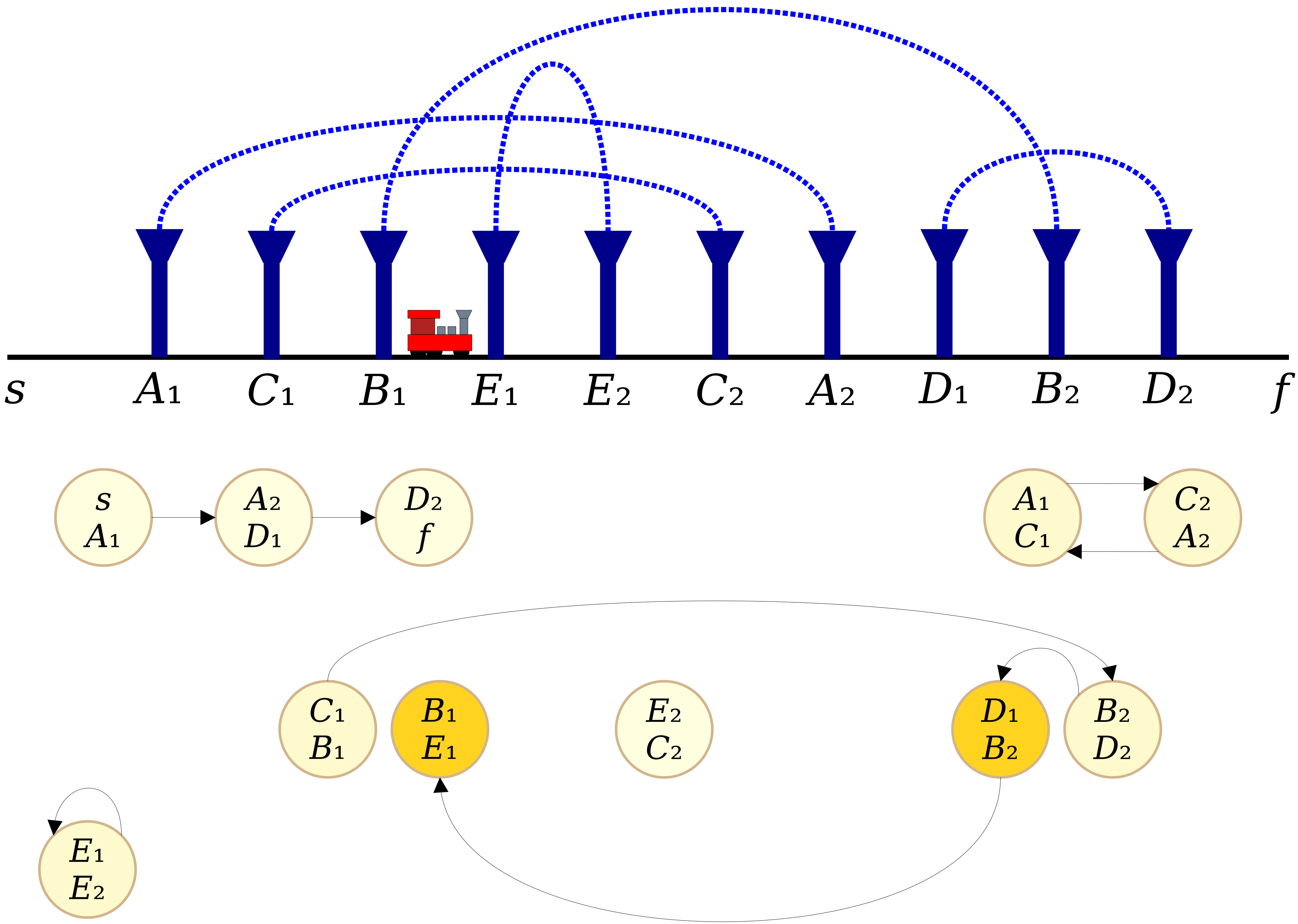
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2

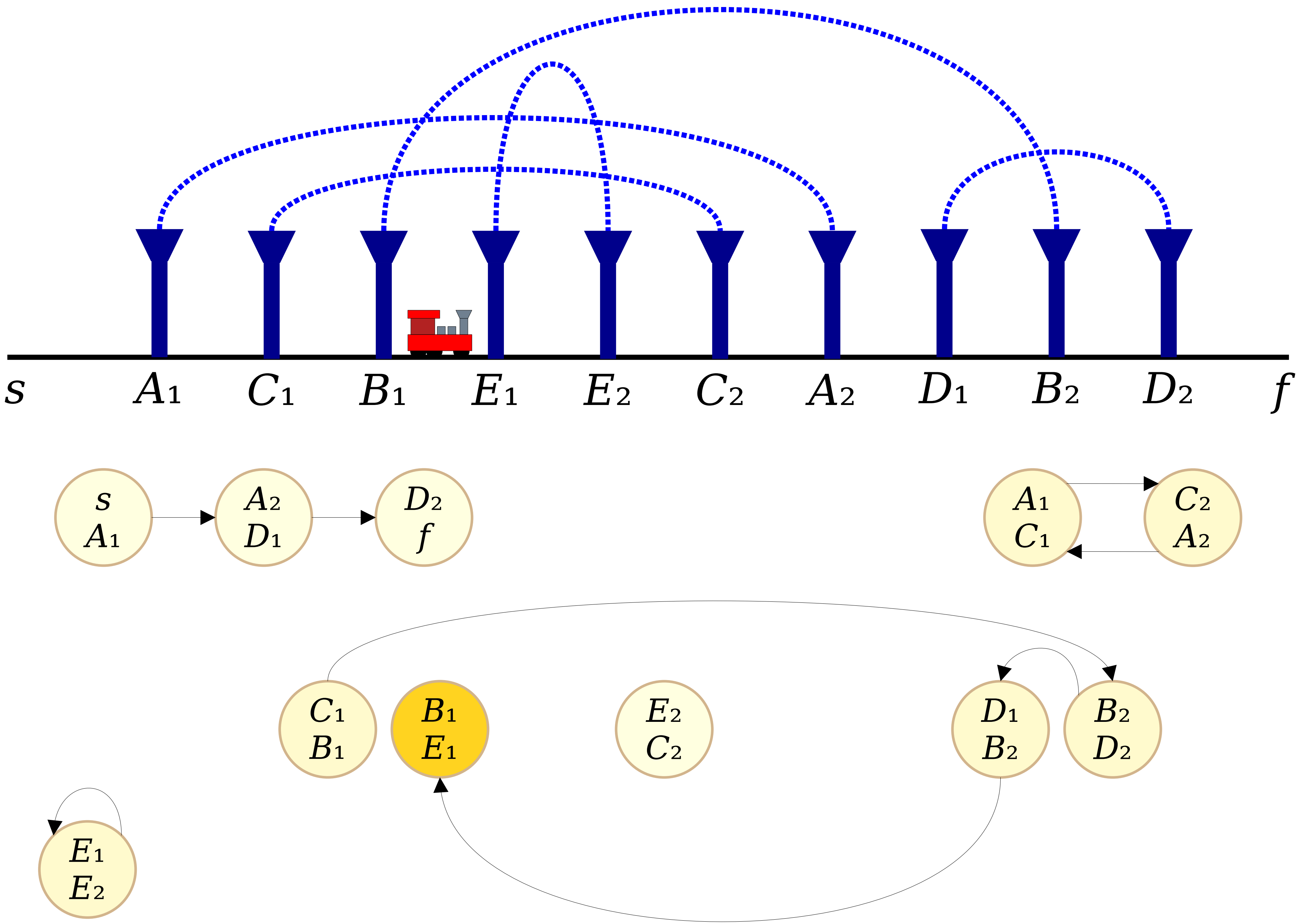


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2

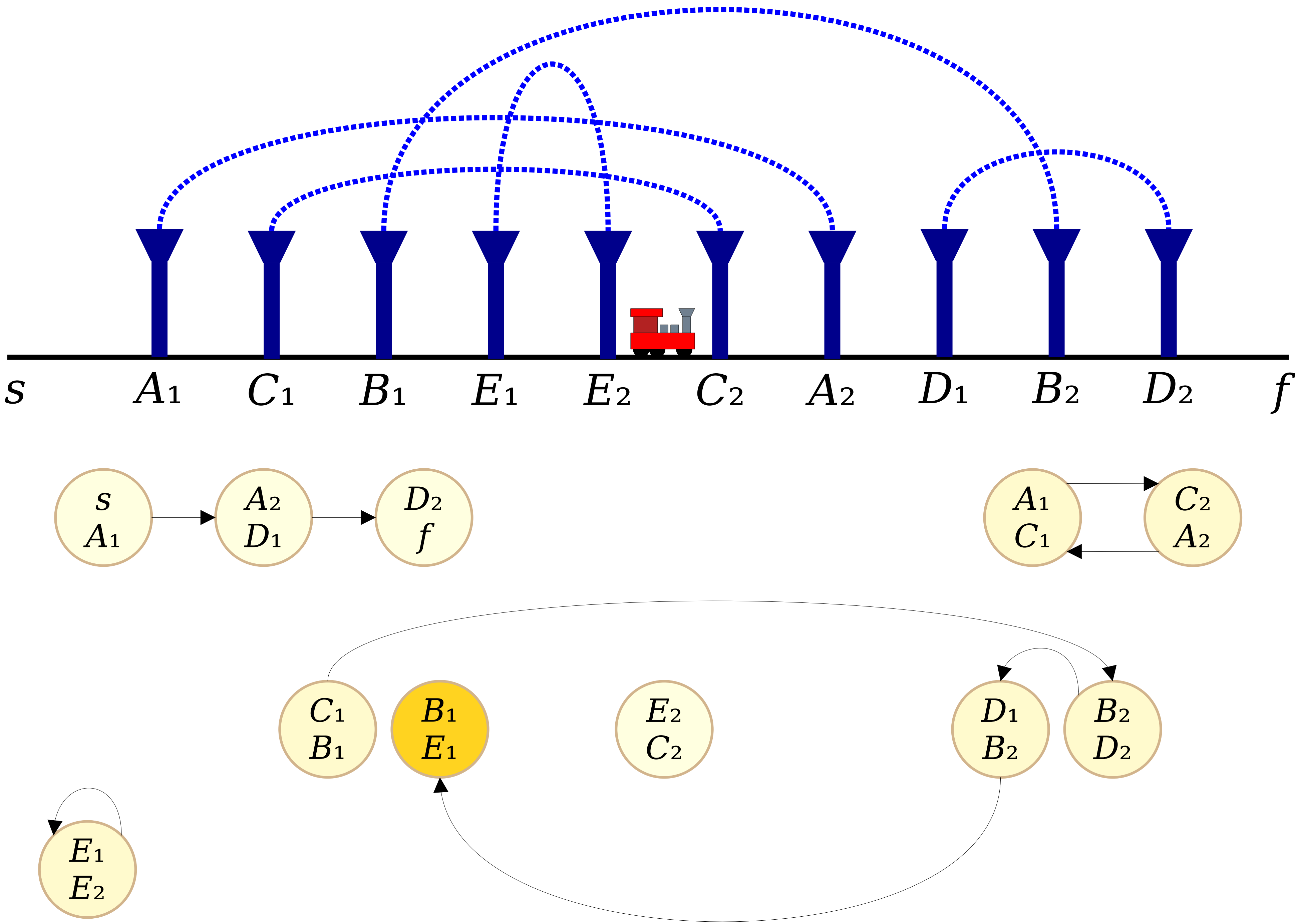


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2

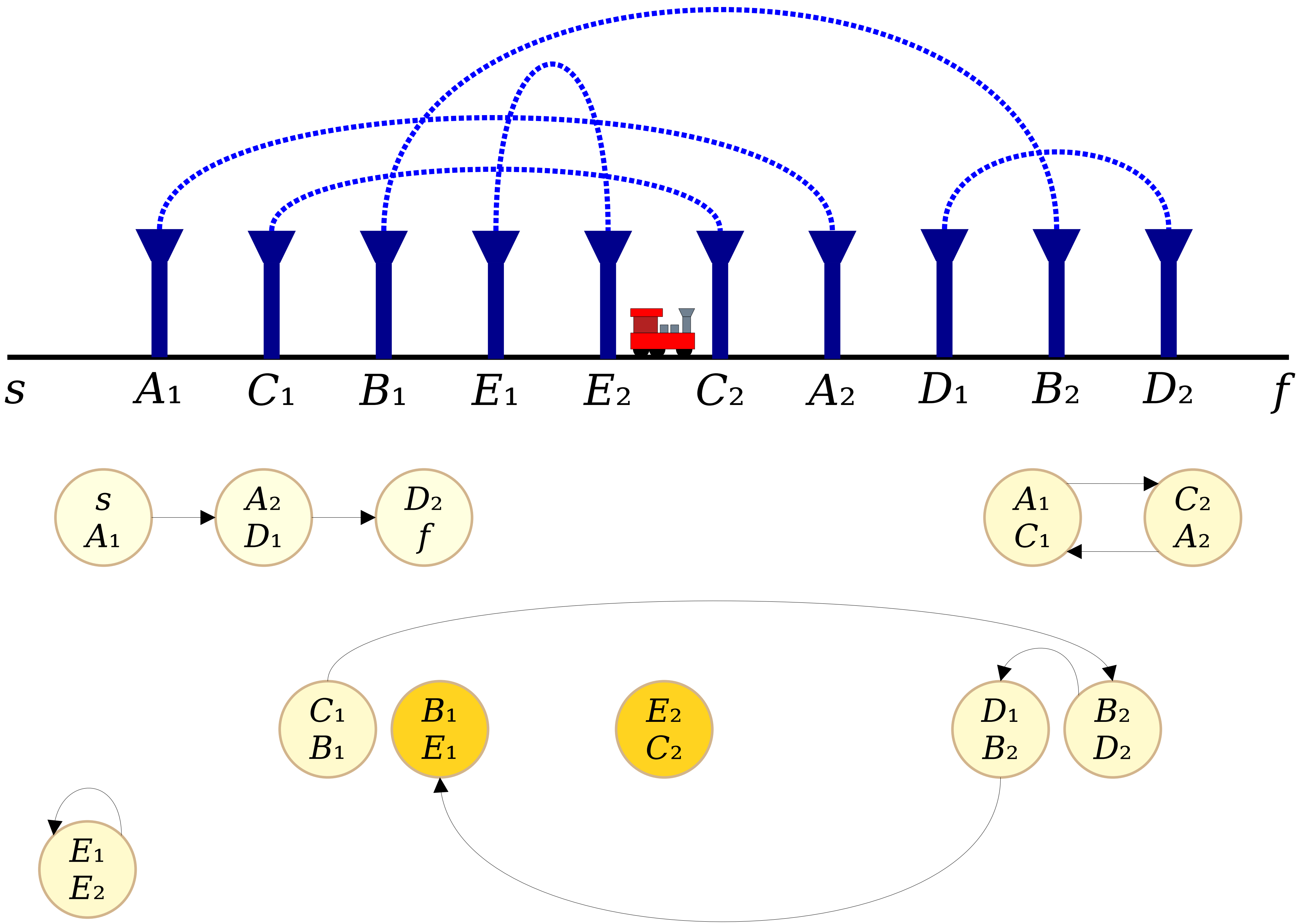

s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2



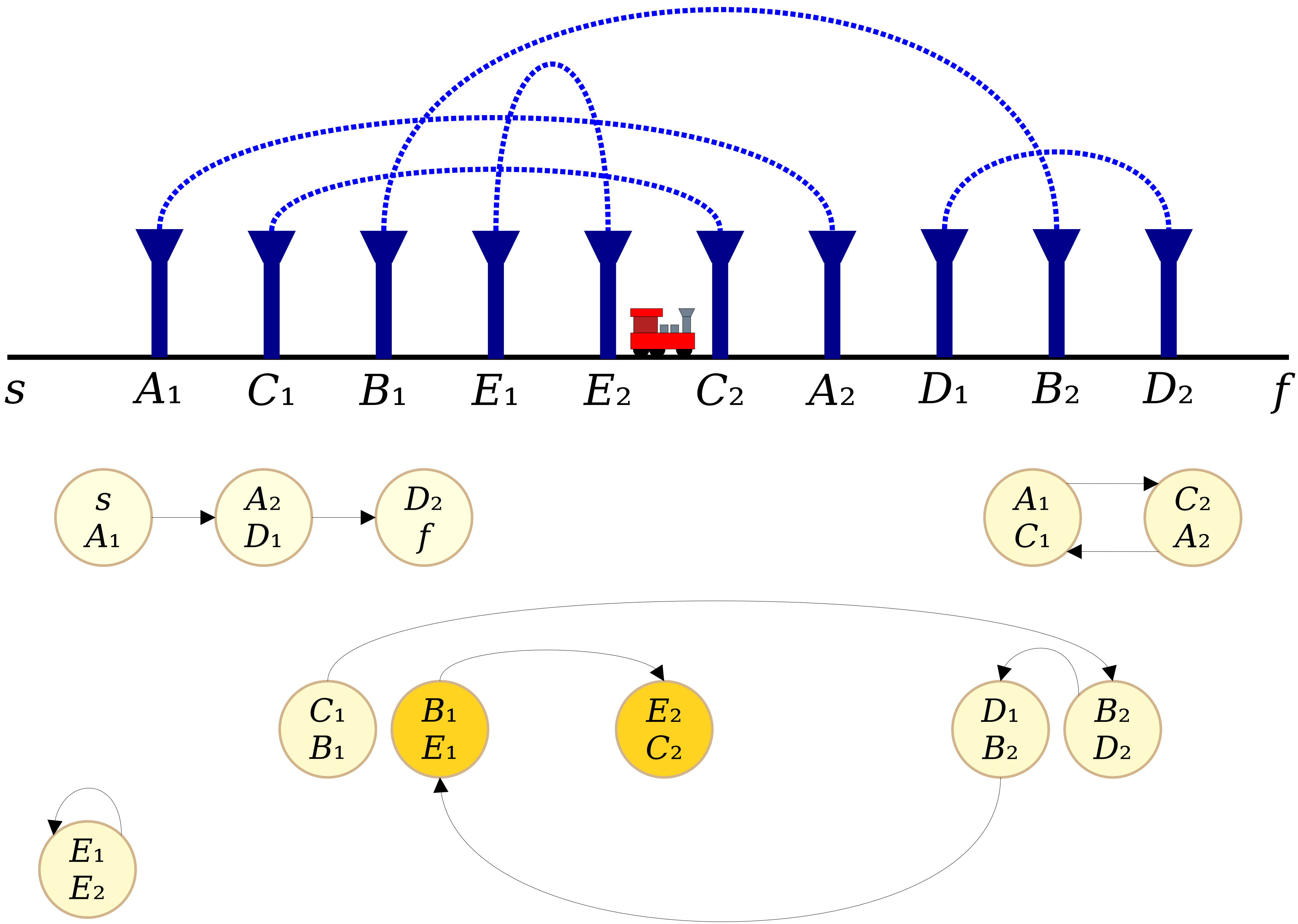
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2



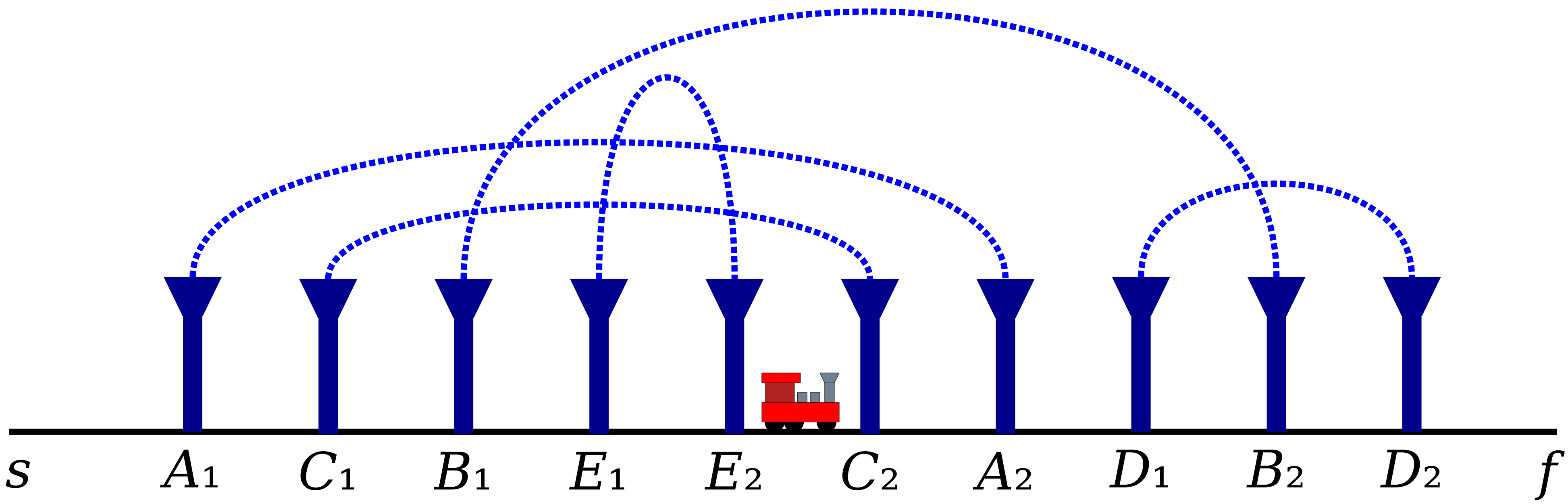
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2



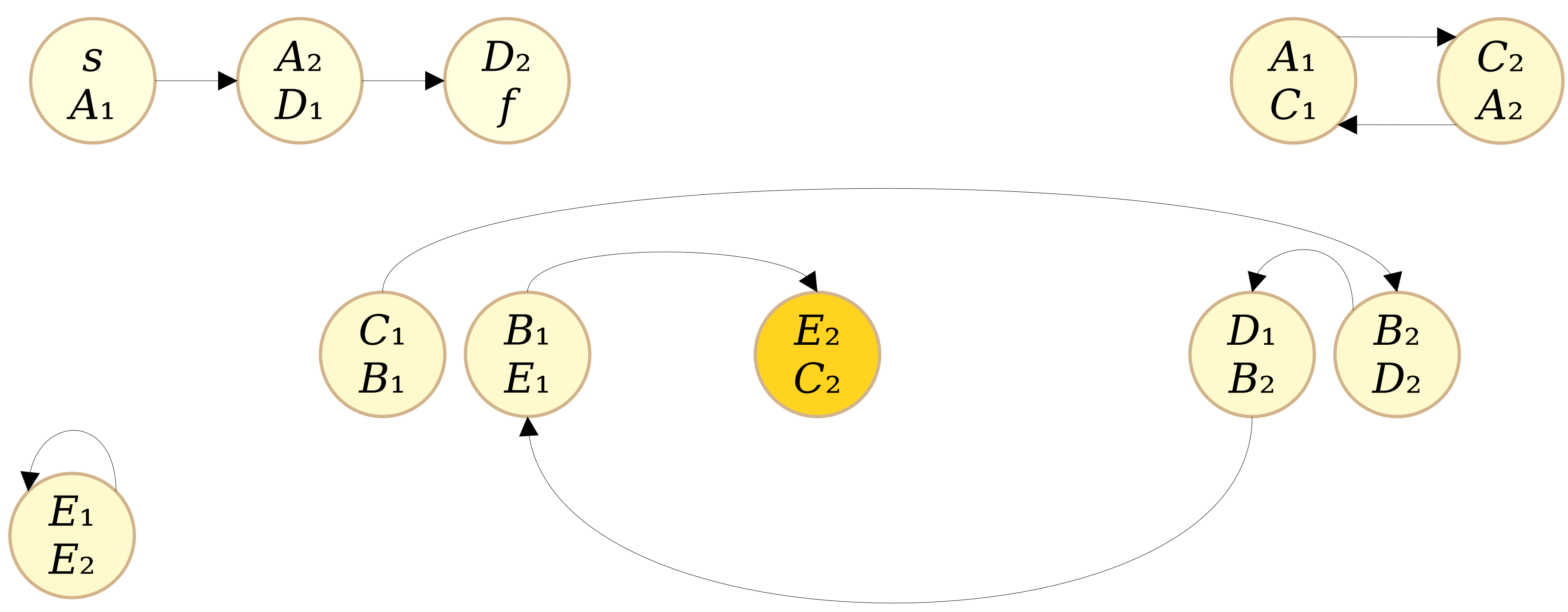
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2



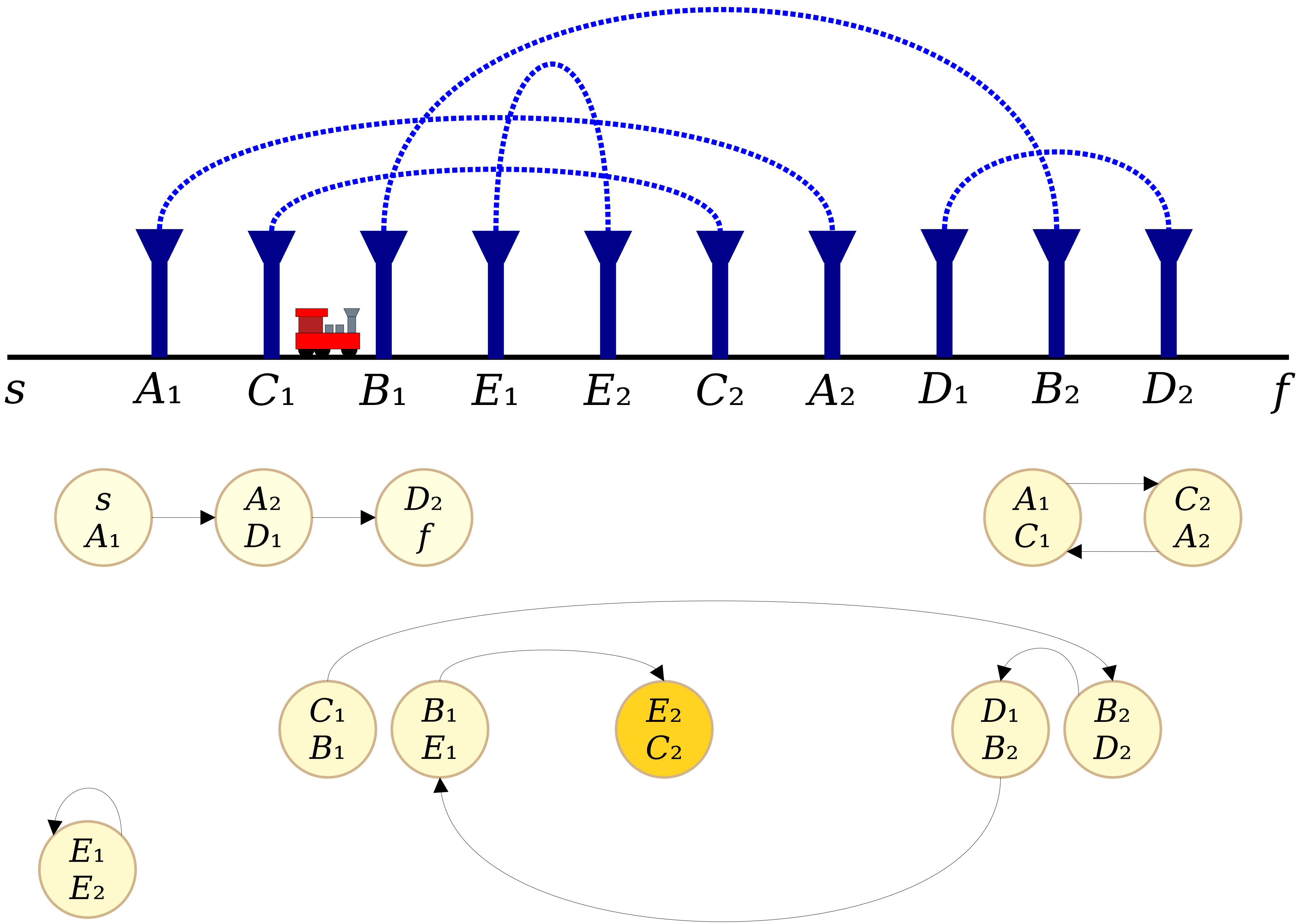
s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_2 A_2
C_1 B_1
B_1 E_1
E_2 C_2
D_1 B_2
B_2 D_2
E_1 E_2



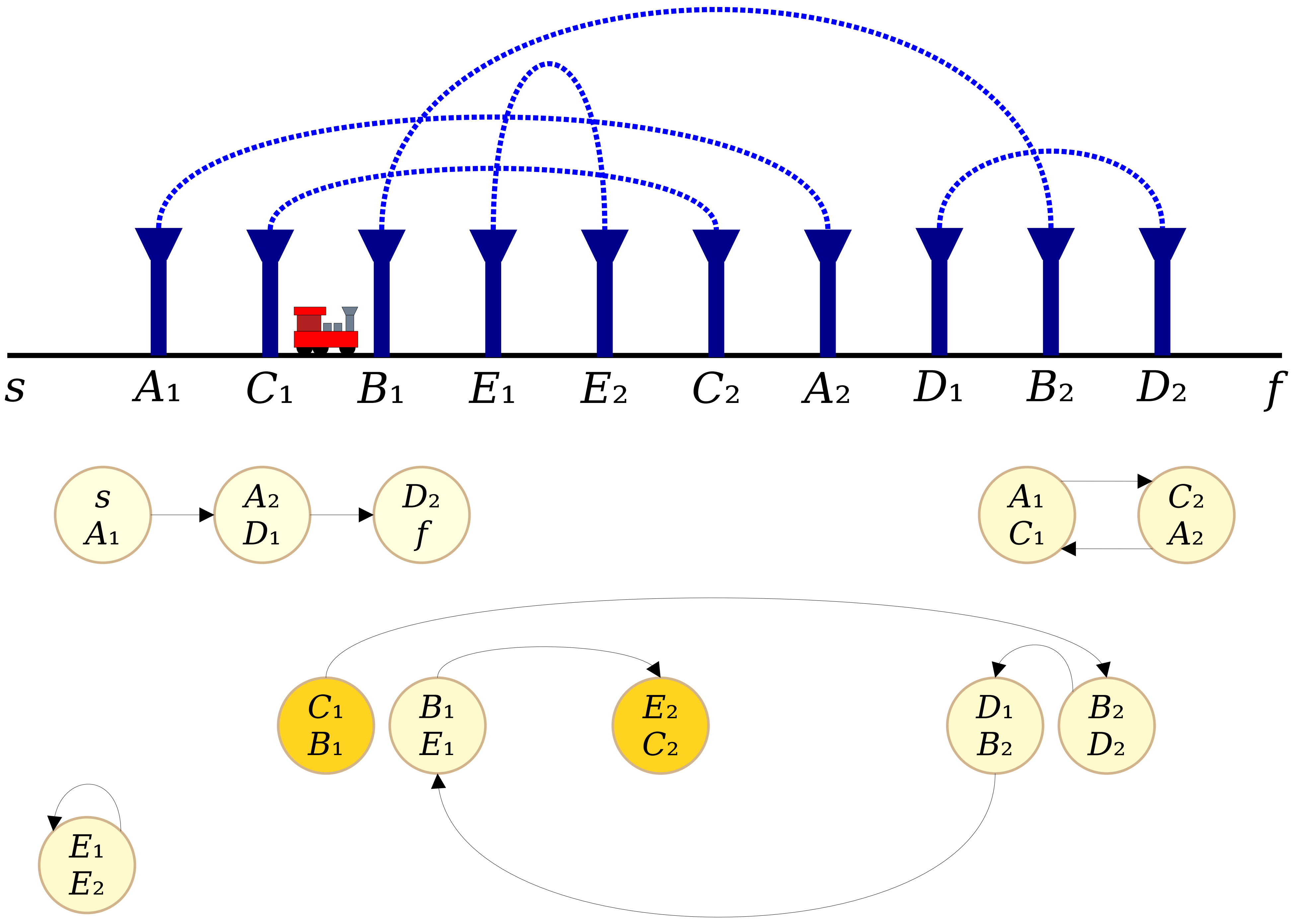
s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f



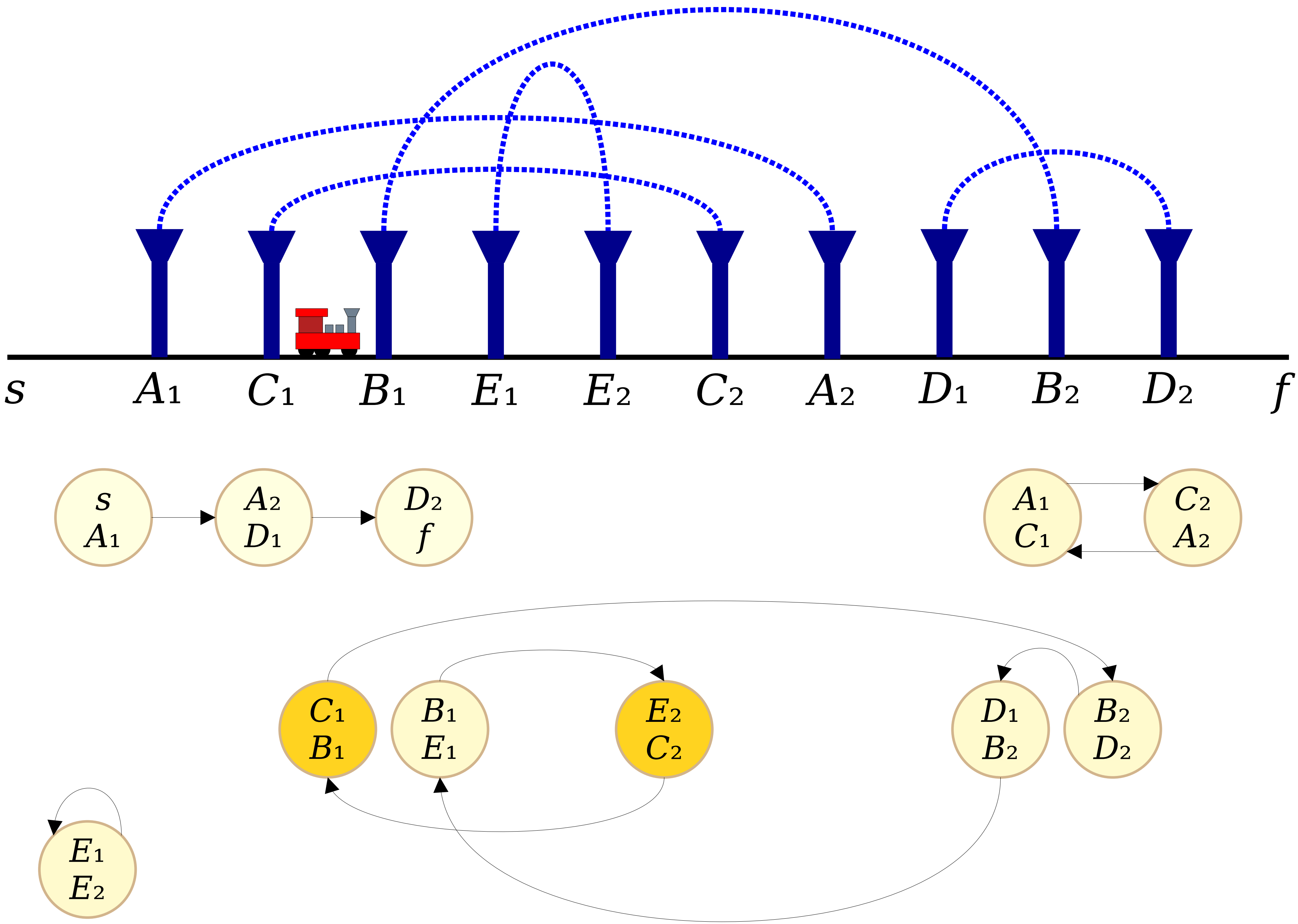
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_2 A_2
C_1 B_1
B_1 E_1
E_2 C_2
D_1 B_2
B_2 D_2
E_1 E_2



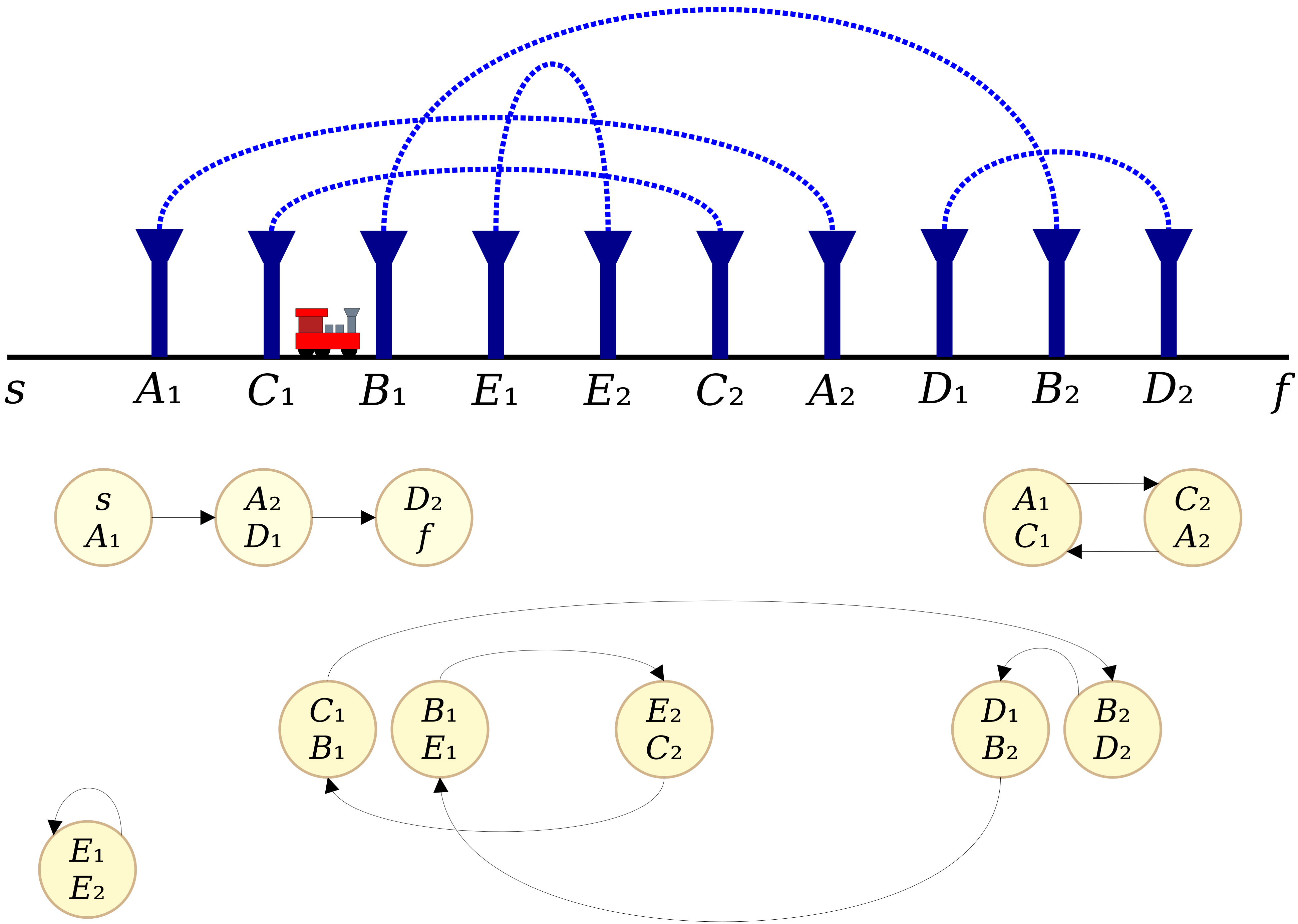
s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2

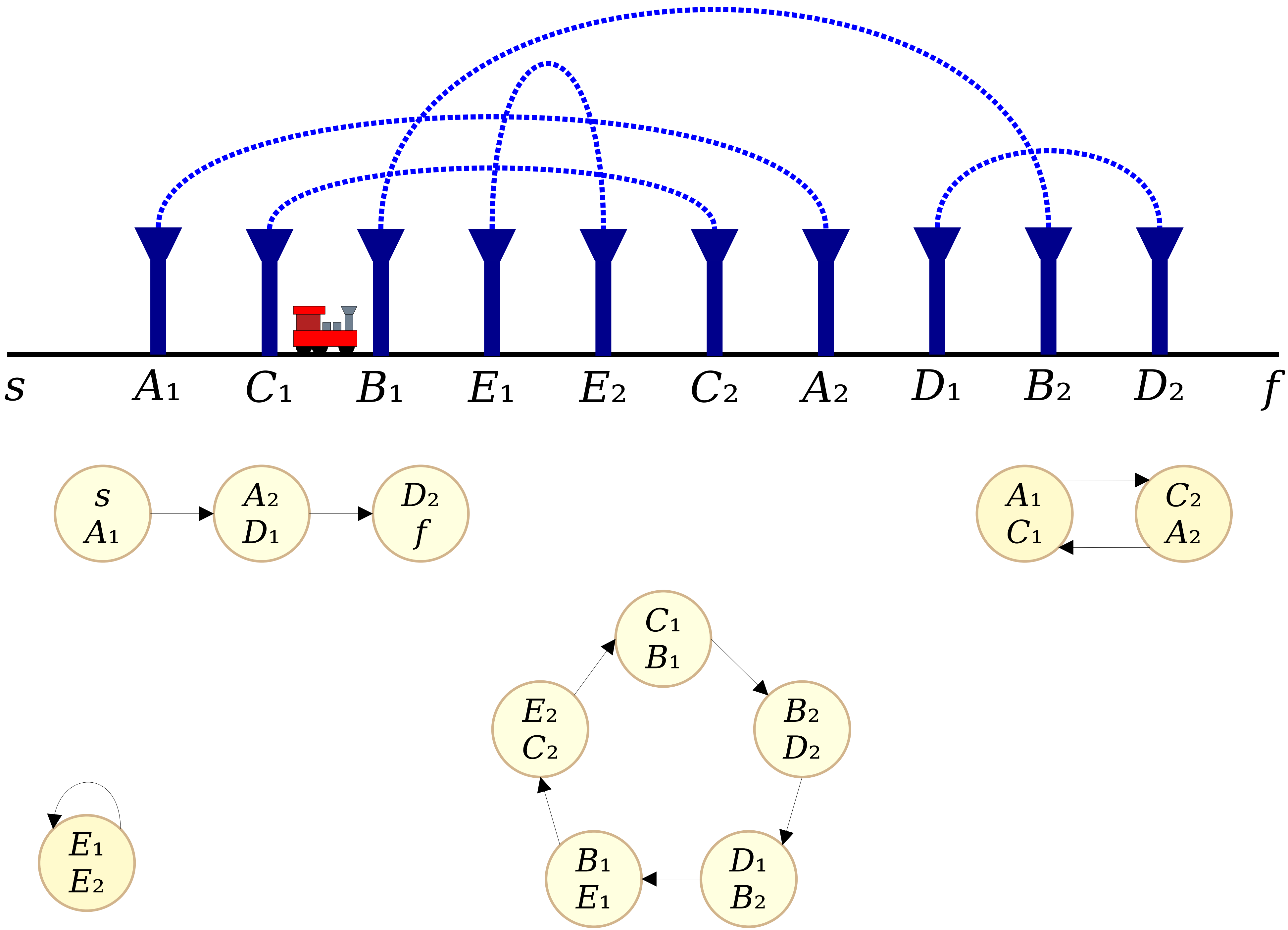


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2



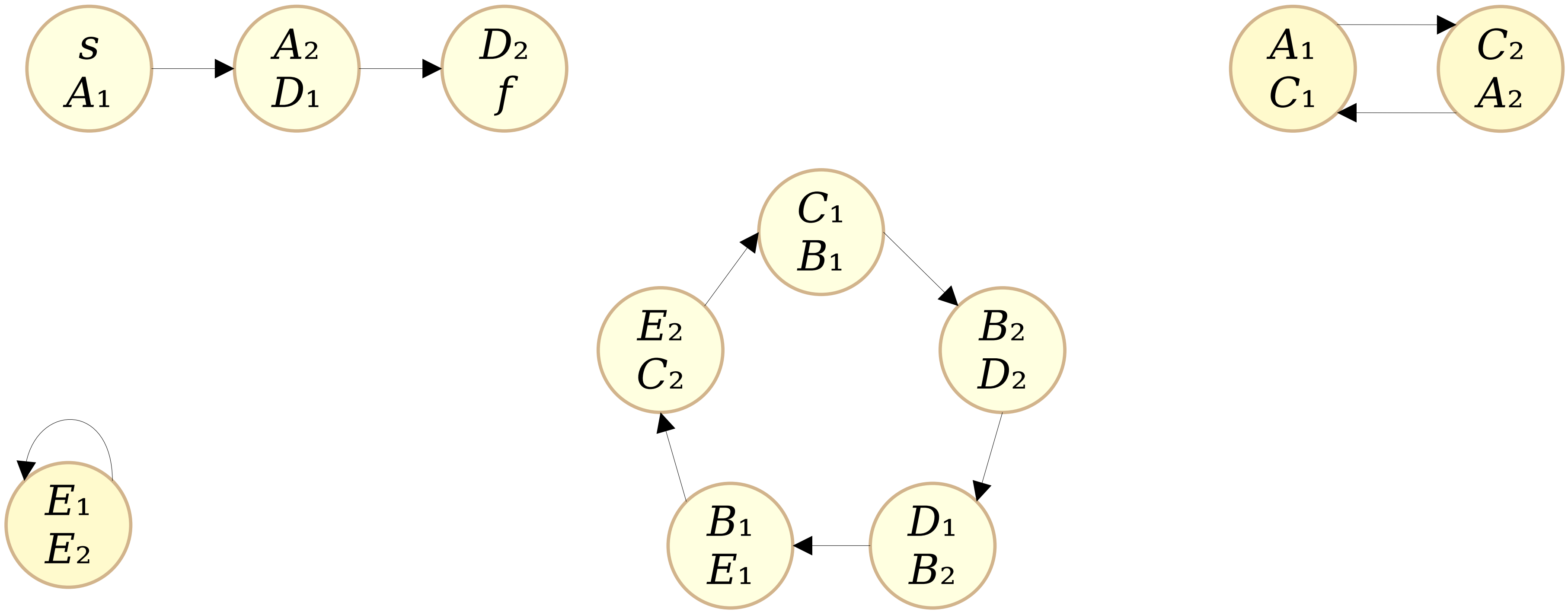
s
A_1
C_1
B_1
E_1
E_2
C_2
A_2
D_1
B_2
D_2
f
s A_1
A_2 D_1
D_2 f
A_1 C_1
C_2 A_2
C_1 B_1
B_1 E_1
E_2 C_2
D_1 B_2
B_2 D_2
E_1 E_2


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
B1 E1
E2 C2
D1 B2
B2 D2
E1 E2


s
A1
C1
B1
E1
E2
C2
A2
D1
B2
D2
f
s A1
A2 D1
D2 f
A1 C1
C2 A2
C1 B1
E2 C2
B2 D2
E1 E2
B1 E1
D1 B2

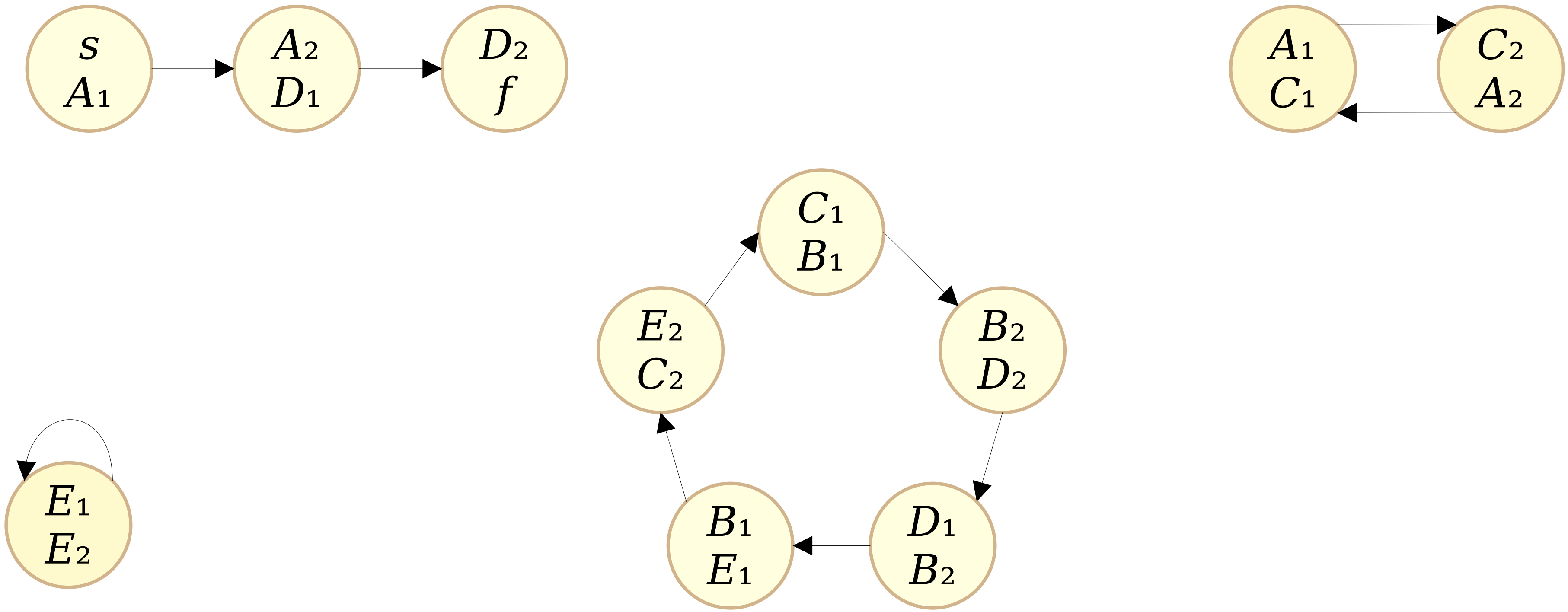
The Teleporter Digraph

- Each line of teleporters gives rise to a directed graph.
  - Each node in the graph represents a segment.
  - Each edge represents following a teleporter.
- That digraph consists of paths and cycles.
- ***Question:*** Why does the digraph look like this?



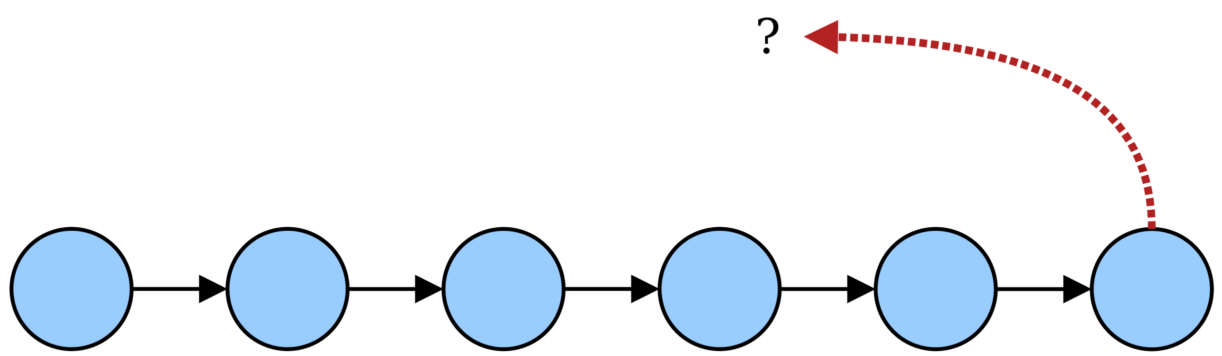
The Teleporter Digraph

- In a directed graph, the ***indegree*** of a node is the number of edges entering that node. The ***outdegree*** of a node is the number of edges leaving that node.
- Notice anything about the indegrees and outdegrees of this digraph?



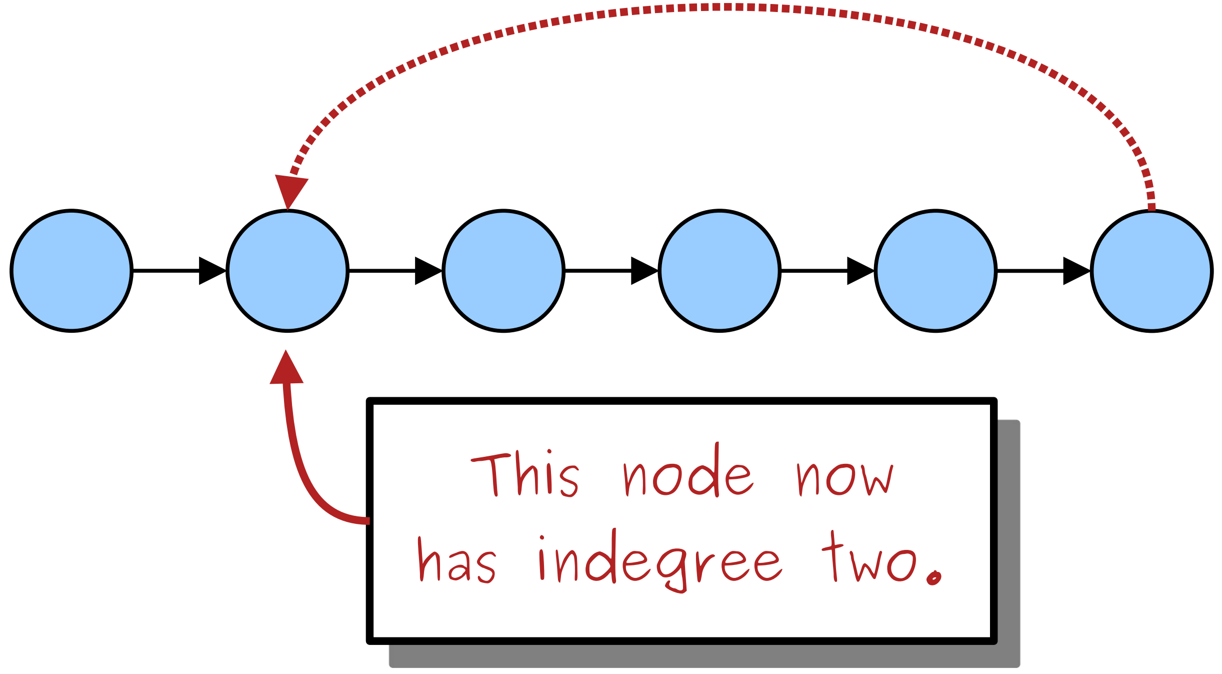
The Teleporter Digraph

- Let $G = (V, E)$ be a digraph where each node's indegree is at most one and each node's outdegree is at most one.
- ***Theorem:*** Any walk starting at a node of indegree zero is also a path.



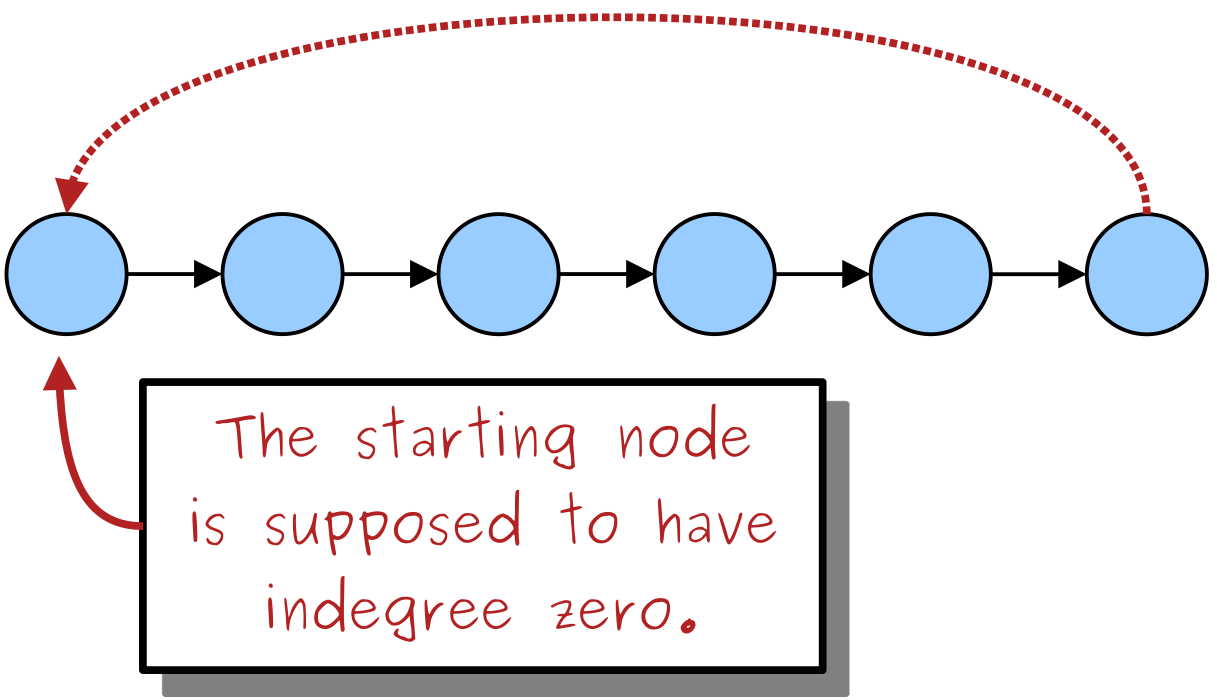
The Teleporter Digraph

- Let $G = (V, E)$ be a digraph where each node's indegree is at most one and each node's outdegree is at most one.
- ***Theorem:*** Any walk starting at a node of indegree zero is also a path.



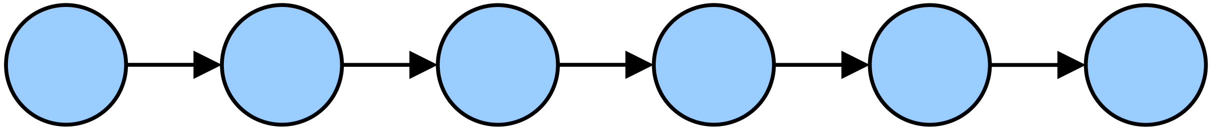
The Teleporter Digraph

- Let $G = (V, E)$ be a digraph where each node's indegree is at most one and each node's outdegree is at most one.
- ***Theorem:*** Any walk starting at a node of indegree zero is also a path.



The Teleporter Digraph

- Let $G = (V, E)$ be a digraph where each node's indegree is at most one and each node's outdegree is at most one.
- ***Theorem:*** Any walk starting at a node of indegree zero is also a path.



Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof:

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node.

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node.

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node. Label the nodes found this way as

$$v_0, v_1, v_2, v_3, \ldots, v_k.$$

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node. Label the nodes found this way as

$$v_0, v_1, v_2, v_3, \ldots, v_k.$$

Nodes v_0, v_1, …, and v_k are distinct because we've stopped just before revisiting a node.

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node. Label the nodes found this way as

$$v_0, v_1, v_2, v_3, \ldots, v_k.$$

Nodes v_0, v_1, …, and v_k are distinct because we've stopped just before revisiting a node. We also know that the next node in the walk (call it r) is a repeated node, with (v_k, r) being a directed edge in E.

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node. Label the nodes found this way as

$$v_0, v_1, v_2, v_3, \ldots, v_k.$$

Nodes $v_0, v_1, \ldots,$ and v_k are distinct because we've stopped just before revisiting a node. We also know that the next node in the walk (call it r) is a repeated node, with (v_k, r) being a directed edge in E. We now ask: which earlier node is r equal to?

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node. Label the nodes found this way as

$$v_0, v_1, v_2, v_3, \ldots, v_k.$$

Nodes $v_0, v_1, \ldots,$ and v_k are distinct because we've stopped just before revisiting a node. We also know that the next node in the walk (call it r) is a repeated node, with (v_k, r) being a directed edge in E. We now ask: which earlier node is r equal to?

Case 1: $r = v_0$.

Case 2: $r = v_i$ for some $i \neq 0$.

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node. Label the nodes found this way as

$$v_0, v_1, v_2, v_3, \ldots, v_k.$$

Nodes $v_0, v_1, \ldots,$ and v_k are distinct because we've stopped just before revisiting a node. We also know that the next node in the walk (call it r) is a repeated node, with (v_k, r) being a directed edge in E. We now ask: which earlier node is r equal to?

Case 1: $r = v_0$. This means that (v_k, v_0) is a directed edge, which is impossible because v_0 has indegree zero.

Case 2: $r = v_i$ for some $i \neq 0$.

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node. Label the nodes found this way as

$$v_0, v_1, v_2, v_3, \ldots, v_k.$$

Nodes $v_0, v_1, \ldots,$ and v_k are distinct because we've stopped just before revisiting a node. We also know that the next node in the walk (call it r) is a repeated node, with (v_k, r) being a directed edge in E. We now ask: which earlier node is r equal to?

Case 1: $r = v_0$. This means that (v_k, v_0) is a directed edge, which is impossible because v_0 has indegree zero.

Case 2: $r = v_i$ for some $i \neq 0$. Then (v_{i-1}, v_i) and (v_k, v_i) are directed edges in G, which is impossible because v_i has indegree one.

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node. Label the nodes found this way as

$$v_0, v_1, v_2, v_3, \ldots, v_k.$$

Nodes $v_0, v_1, \ldots,$ and v_k are distinct because we've stopped just before revisiting a node. We also know that the next node in the walk (call it r) is a repeated node, with (v_k, r) being a directed edge in E. We now ask: which earlier node is r equal to?

Case 1: $r = v_0$. This means that (v_k, v_0) is a directed edge, which is impossible because v_0 has indegree zero.

Case 2: $r = v_i$ for some $i \neq 0$. Then (v_{i-1}, v_i) and (v_k, v_i) are directed edges in G, which is impossible because v_i has indegree one.

In either case we've reached a contradiction, so our assumption must have been wrong.

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node. Label the nodes found this way as

$$v_0, v_1, v_2, v_3, \ldots, v_k.$$

Nodes $v_0, v_1, \ldots,$ and v_k are distinct because we've stopped just before revisiting a node. We also know that the next node in the walk (call it r) is a repeated node, with (v_k, r) being a directed edge in E. We now ask: which earlier node is r equal to?

Case 1: $r = v_0$. This means that (v_k, v_0) is a directed edge, which is impossible because v_0 has indegree zero.

Case 2: $r = v_i$ for some $i \neq 0$. Then (v_{i-1}, v_i) and (v_k, v_i) are directed edges in G, which is impossible because v_i has indegree one.

In either case we've reached a contradiction, so our assumption must have been wrong. Thus T is a path.

Theorem: Let $G = (V, E)$ be a directed graph where each node has indegree at most one and outdegree at most one. Consider any walk T beginning at a node v_0 of indegree zero. Then T is a path.

Proof: Suppose for the sake of contradiction that T is not a path, meaning that it contains a repeated node. List the nodes in T, stopping just before we list the first repeated node. Label the nodes found this way as

$$v_0, v_1, v_2, v_3, \ldots, v_k.$$

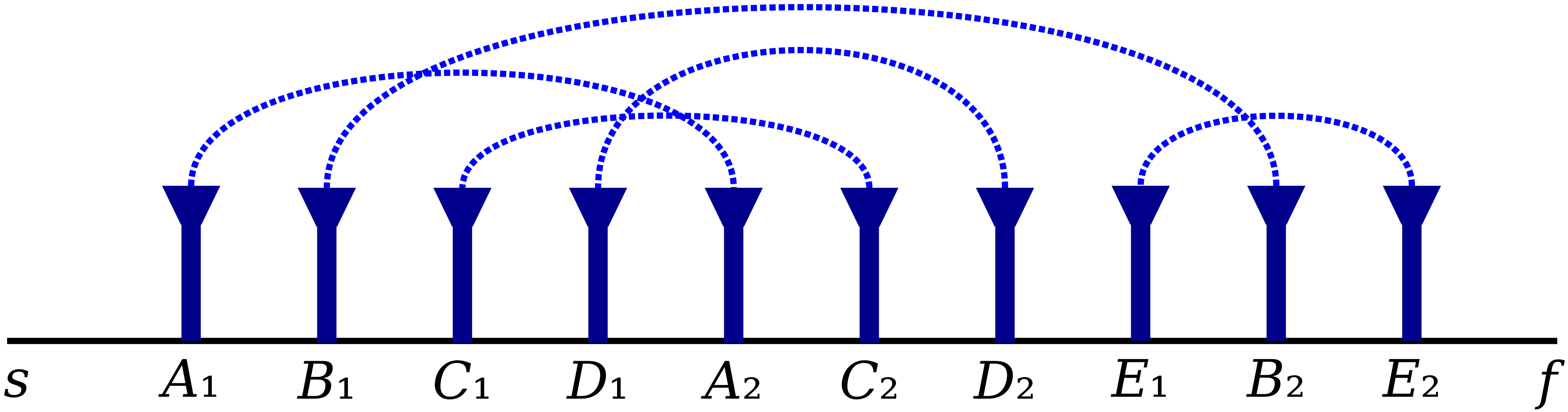
Nodes $v_0, v_1, \ldots,$ and v_k are distinct because we've stopped just before revisiting a node. We also know that the next node in the walk (call it r) is a repeated node, with (v_k, r) being a directed edge in E. We now ask: which earlier node is r equal to?

Case 1: $r = v_0$. This means that (v_k, v_0) is a directed edge, which is impossible because v_0 has indegree zero.

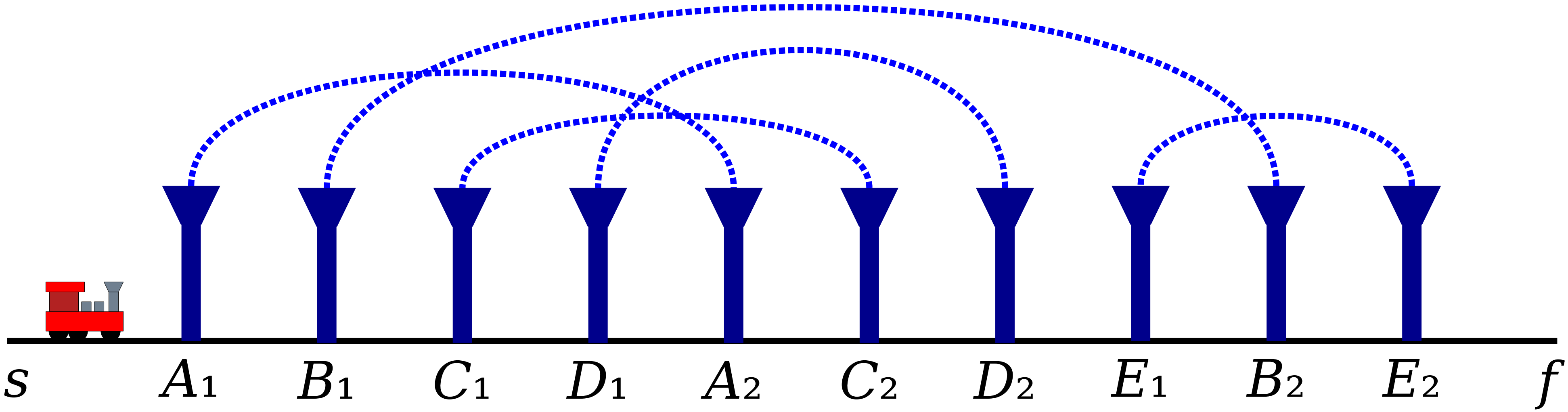
Case 2: $r = v_i$ for some $i \neq 0$. Then (v_{i-1}, v_i) and (v_k, v_i) are directed edges in G, which is impossible because v_i has indegree one.

In either case we've reached a contradiction, so our assumption must have been wrong. Thus T is a path. ■

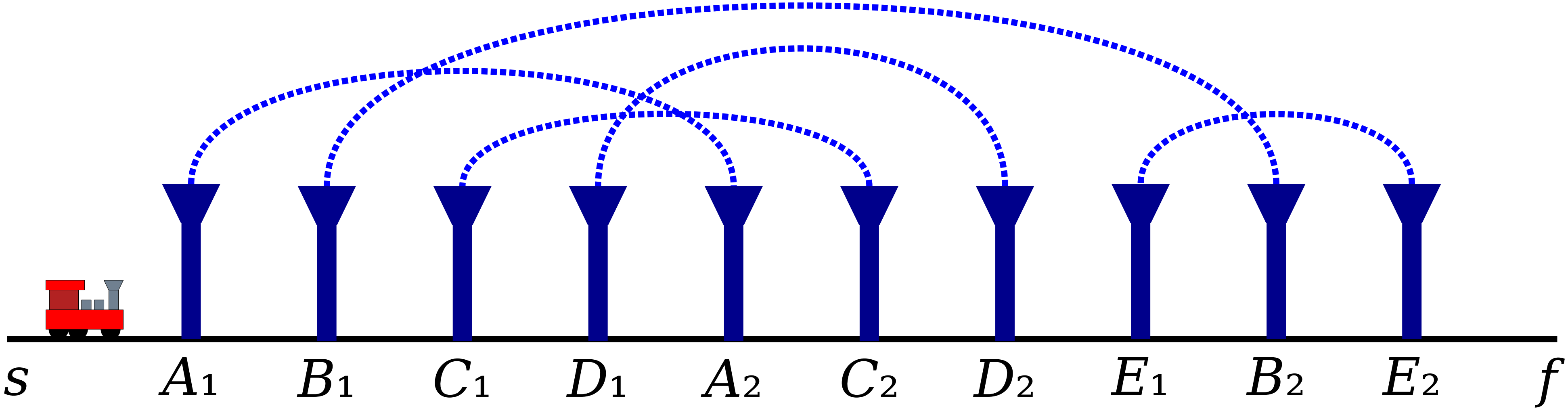
Trapping the Train



Trapping the Train

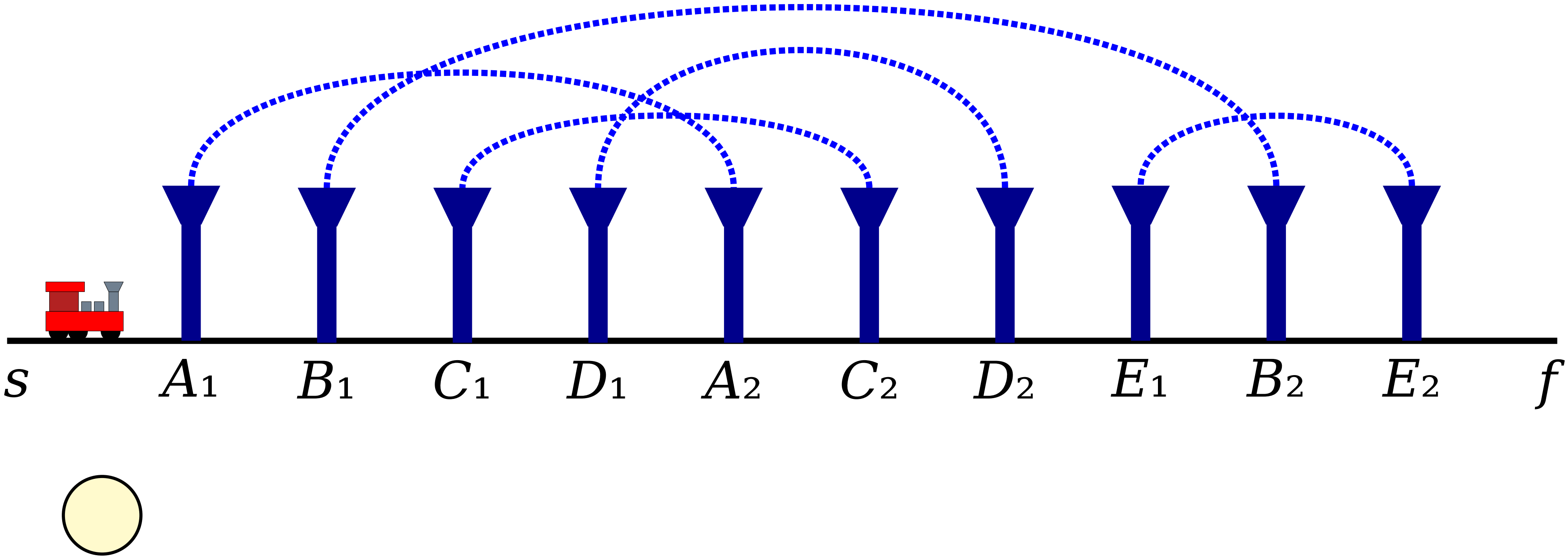


Trapping the Train



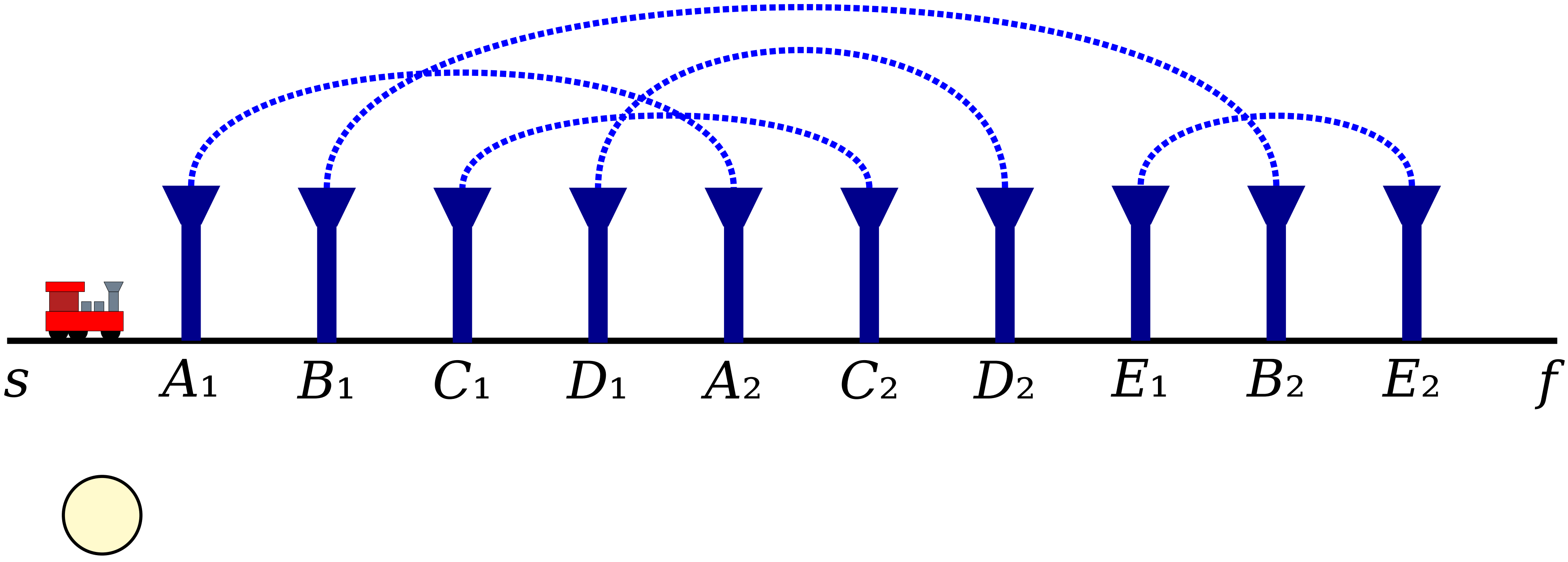
The train begins before the first teleporter, so the start node has indegree zero.

Trapping the Train



The train begins before the first teleporter, so the start node has indegree zero.

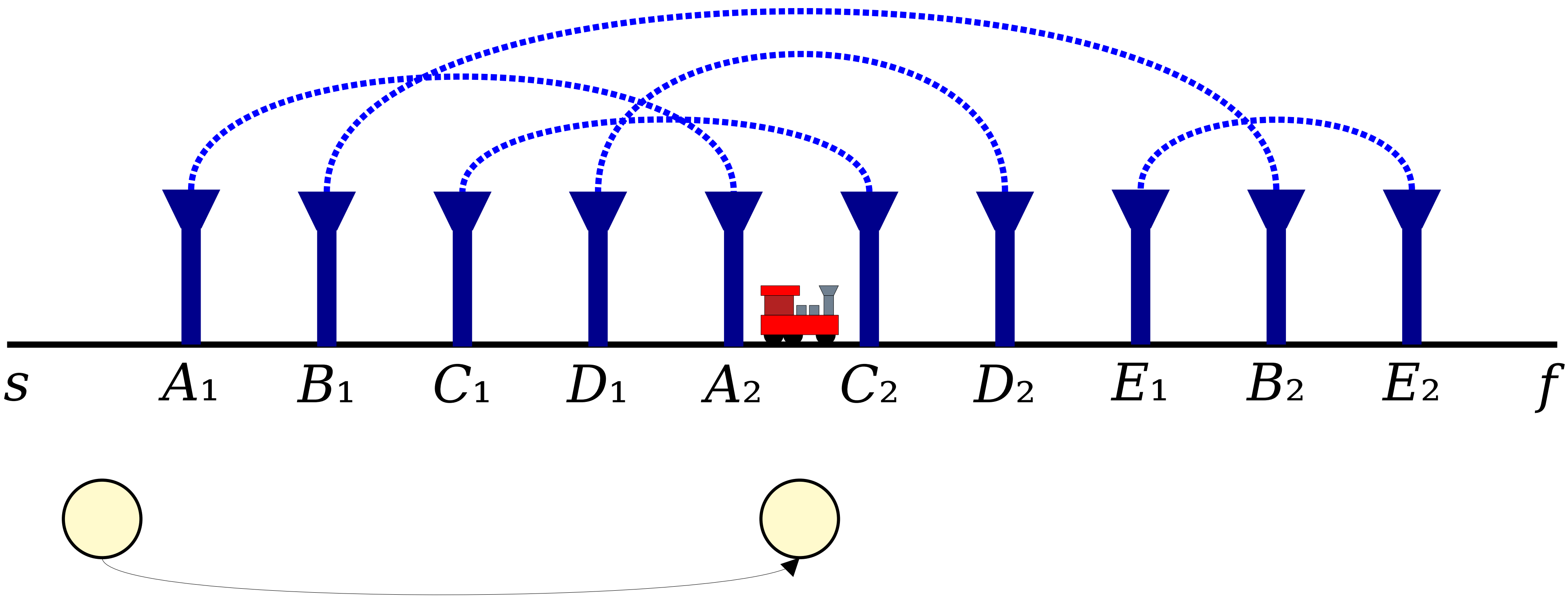
Trapping the Train



The train begins before the first teleporter, so the start node has indegree zero.

Therefore, the walk we trace out is a path, and so it has to end somewhere.

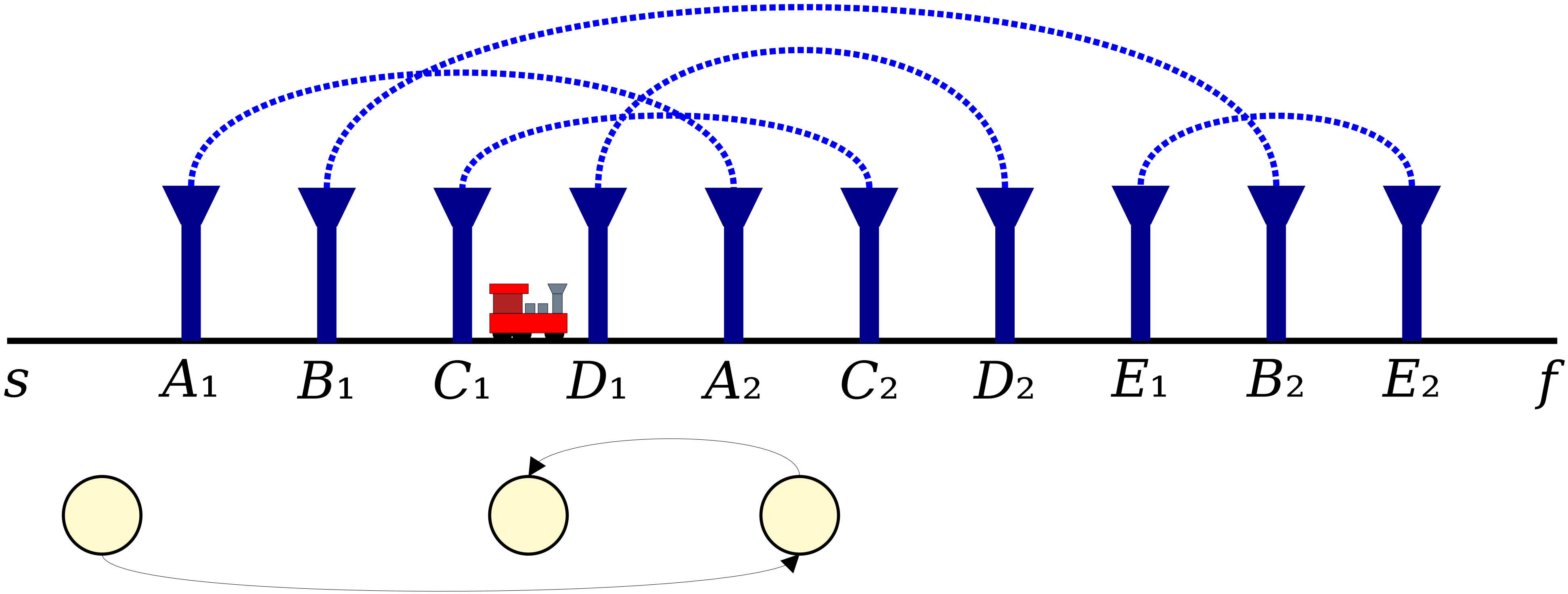
Trapping the Train



The train begins before the first teleporter, so the start node has indegree zero.

Therefore, the walk we trace out is a path, and so it has to end somewhere.

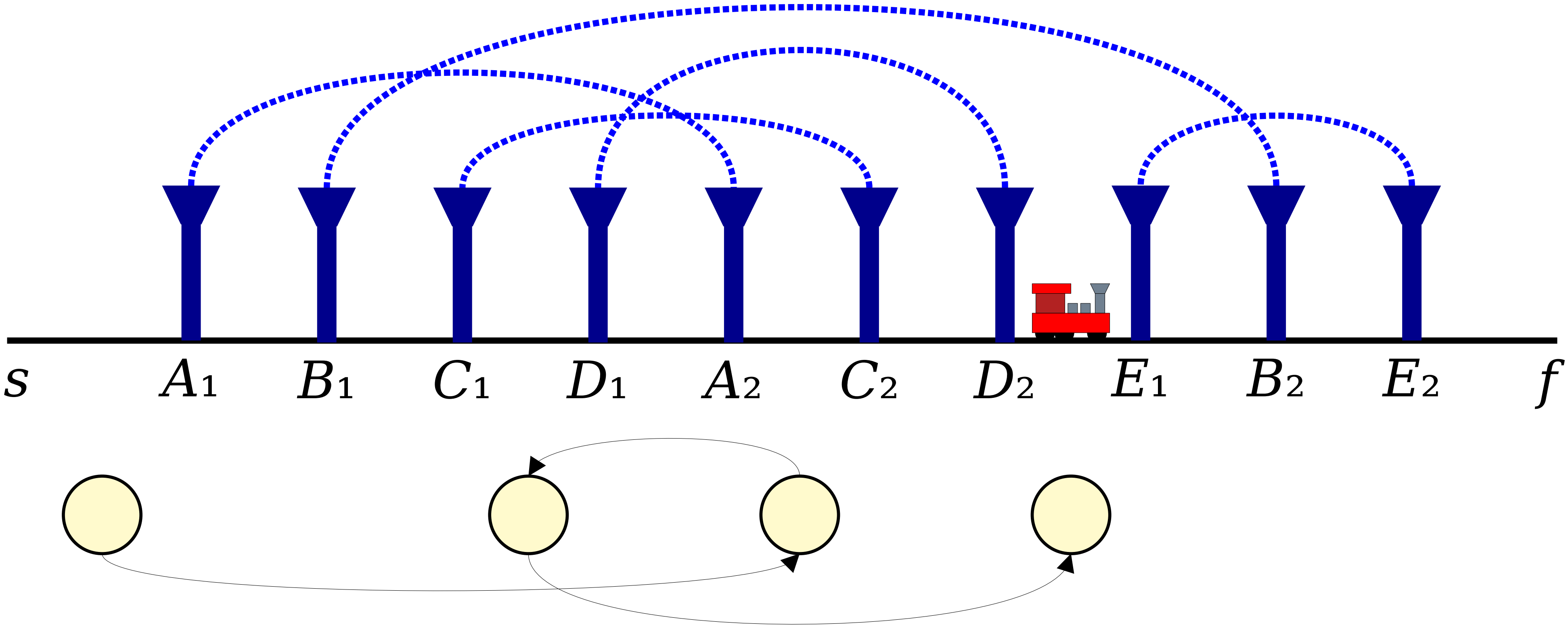
Trapping the Train



The train begins before the first teleporter, so the start node has indegree zero.

Therefore, the walk we trace out is a path, and so it has to end somewhere.

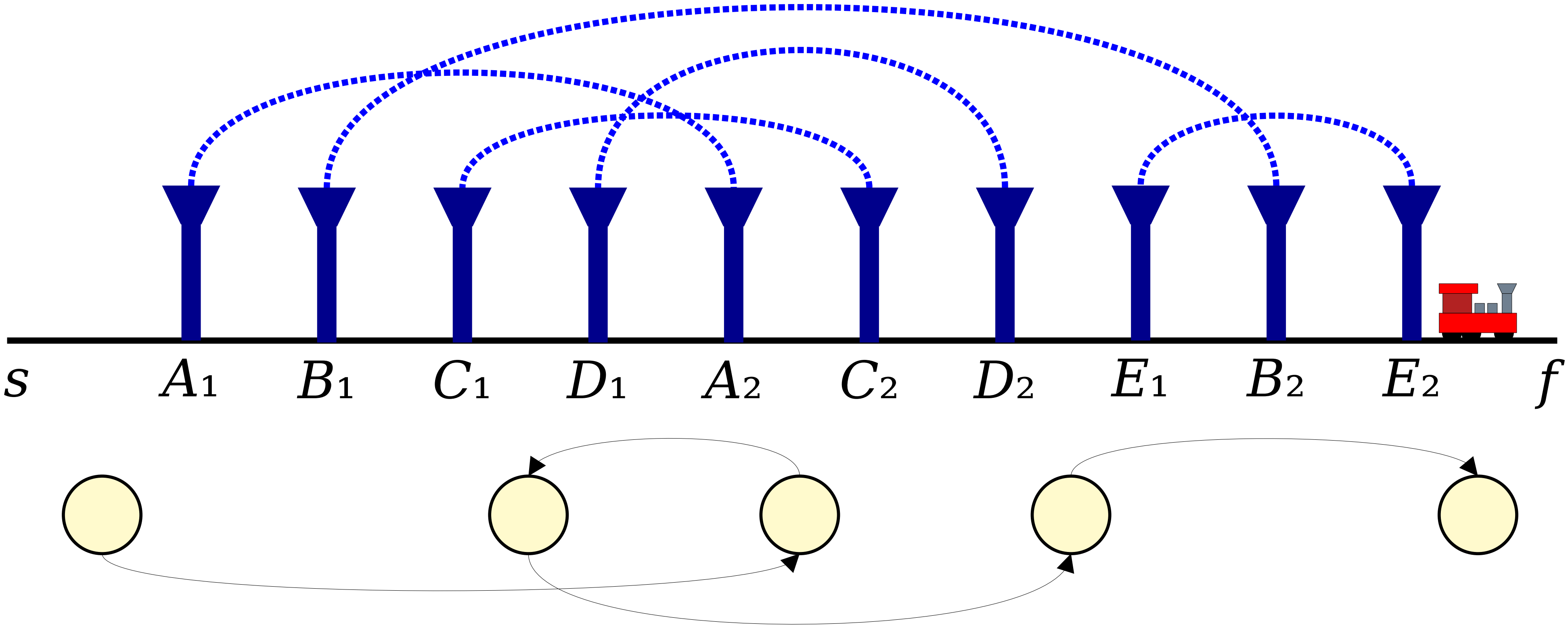
Trapping the Train



The train begins before the first teleporter, so the start node has indegree zero.

Therefore, the walk we trace out is a path, and so it has to end somewhere.

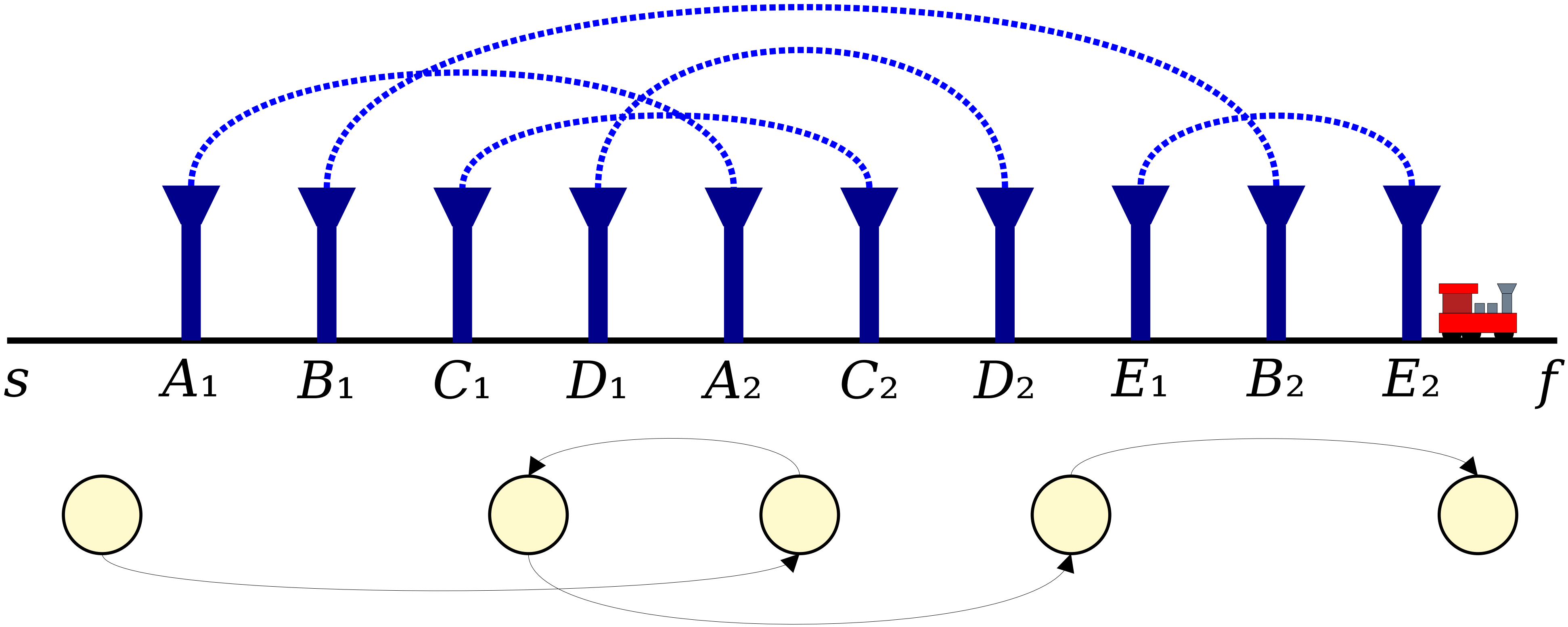
Trapping the Train



The train begins before the first teleporter, so the start node has indegree zero.

Therefore, the walk we trace out is a path, and so it has to end somewhere.

Trapping the Train

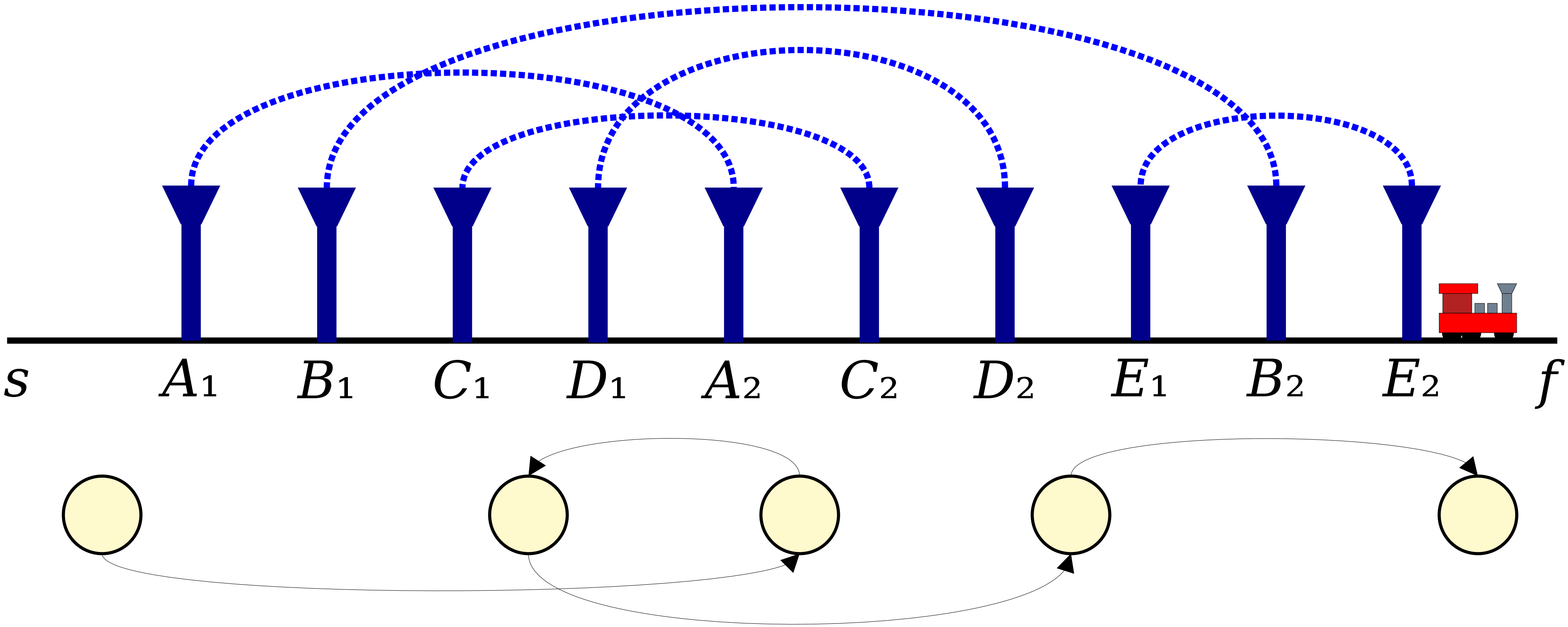


The train begins before the first teleporter, so the start node has indegree zero.

Therefore, the walk we trace out is a path, and so it has to end somewhere.

The only node of outdegree zero is the one after the last teleporter, where the goal is.

Trapping the Train



Theorem: It is impossible to trap the train if it starts before the first teleporter.

Theorem: It is not possible to trap the train in the Teleported Train Problem.

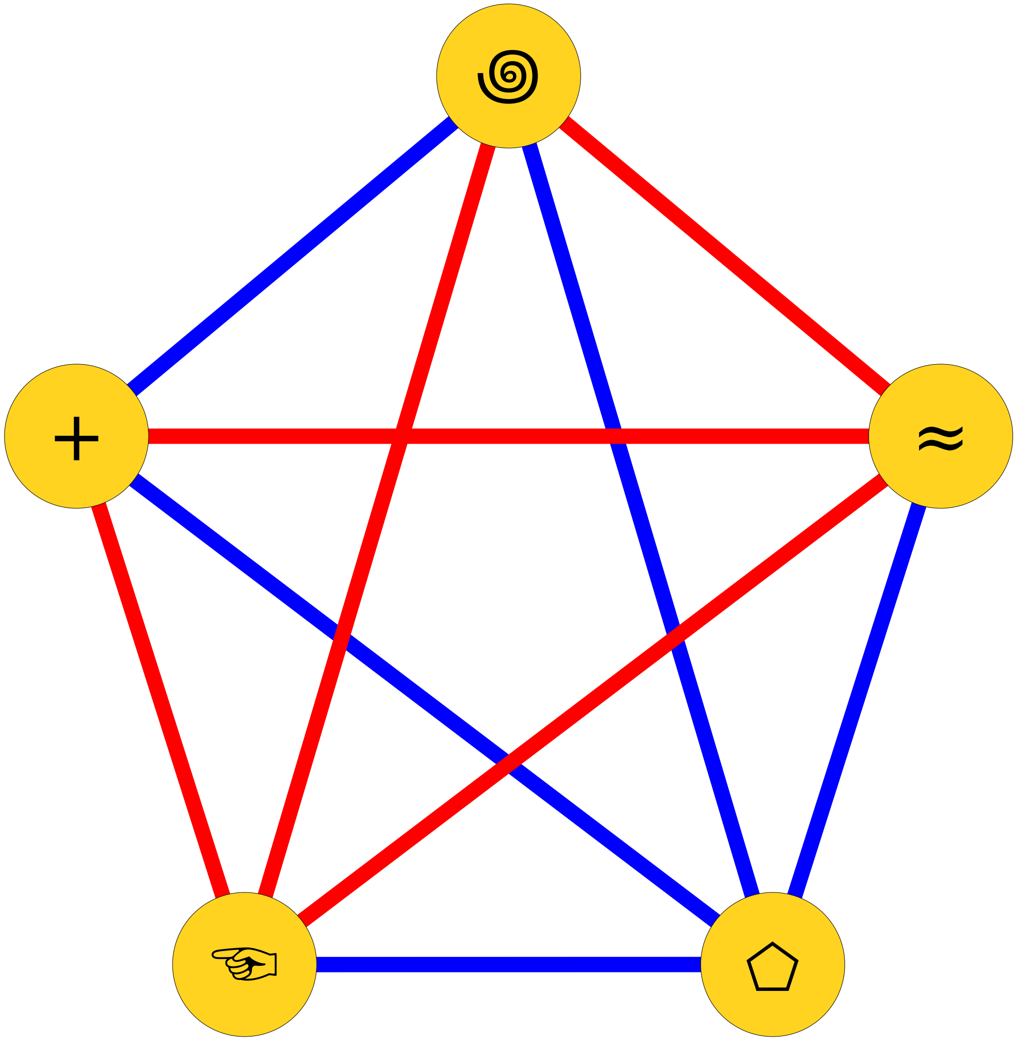
Proof: Consider any arrangement of teleporters. We will prove that the train makes it to the end without getting stuck in a loop.

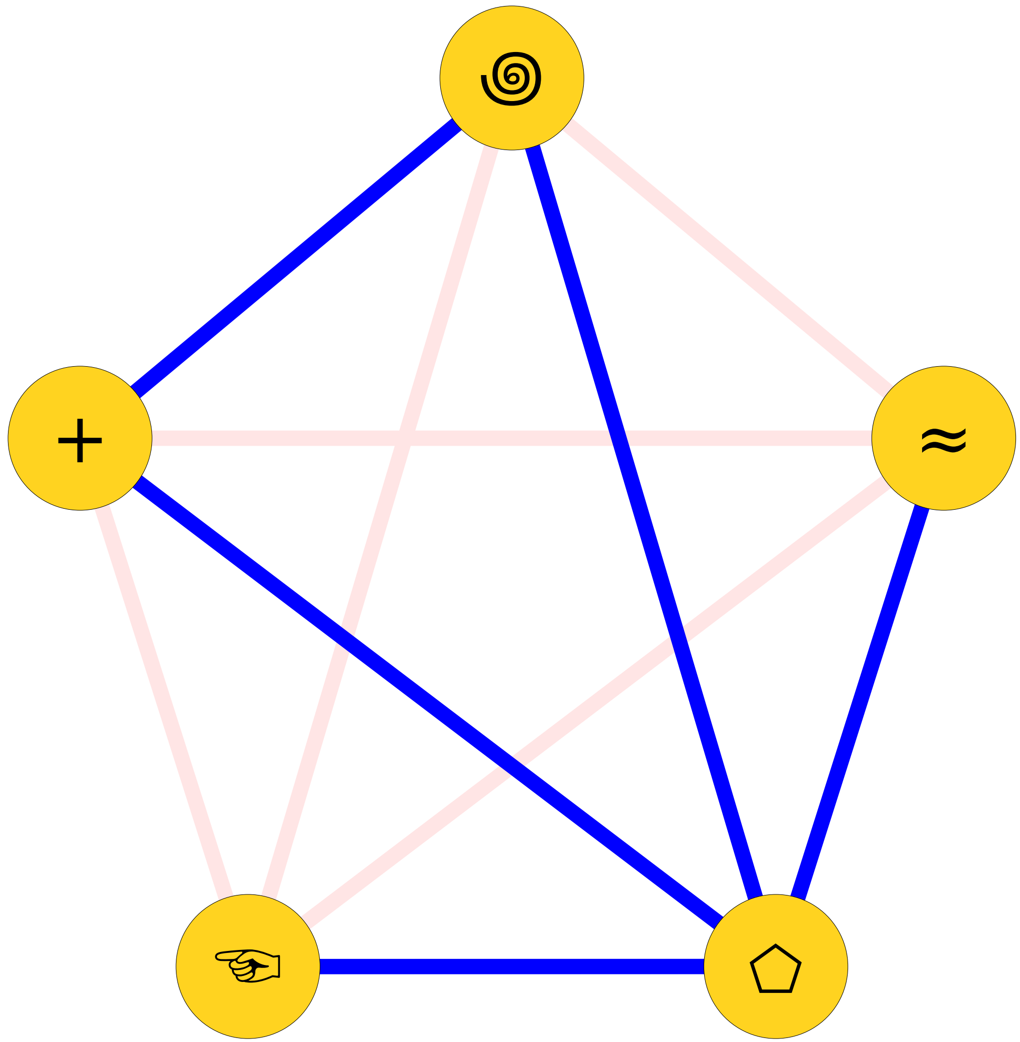
Divide the train track into segments denoting the ranges between two teleporters or between a teleporter and the start/end of the track. From these segments, construct a directed graph whose nodes are the segments and where there's an edge from a segment S_1 to a segment S_2 if, upon reaching the end of segment S_1, the train teleports to the start of segment S_2.

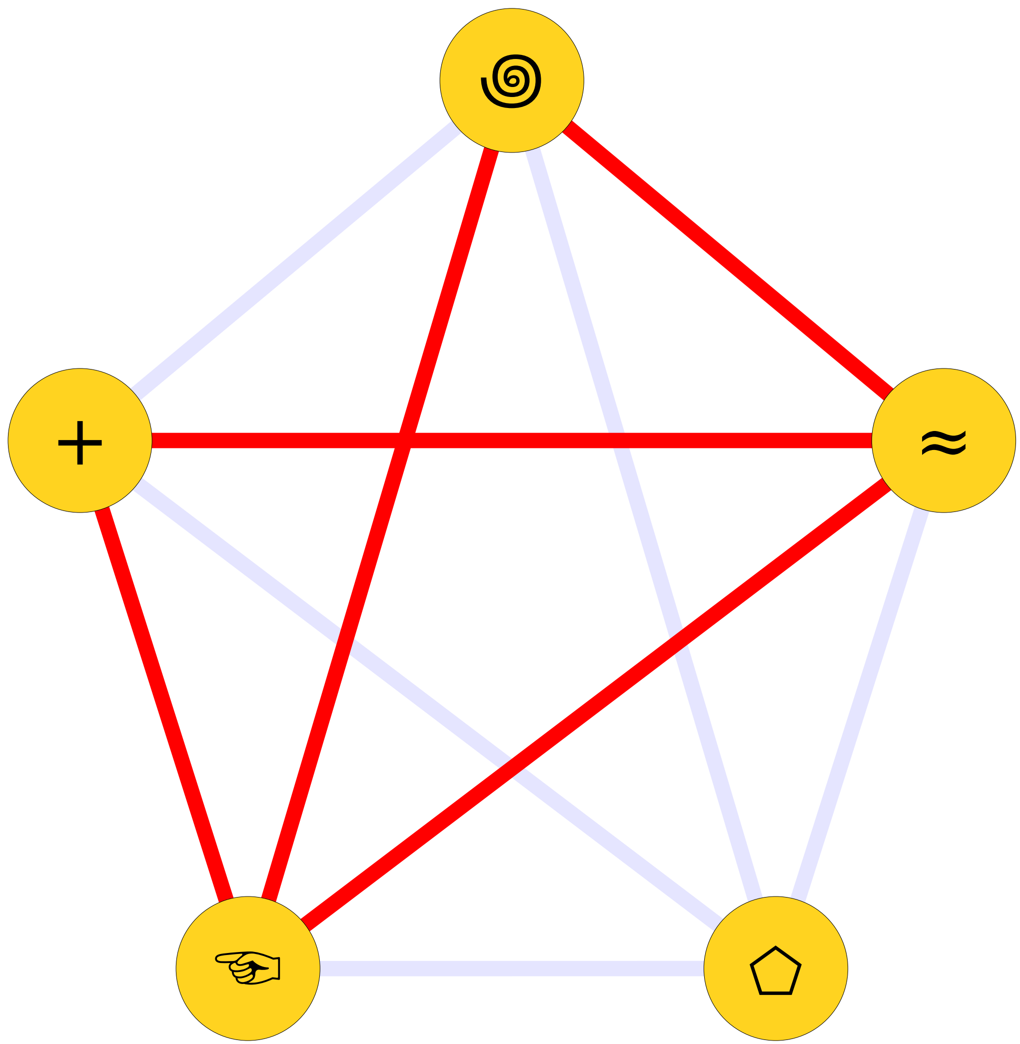
We claim that every node in this graph has indegree at most one and outdegree at most one. To see this, pick any segment. If that segment begins with a teleporter, then it has one incoming edge that originates at the segment that ends with the paired teleporter. If that segment ends with a teleporter, then it has one outgoing edge to the start of the segment with the paired teleporter.

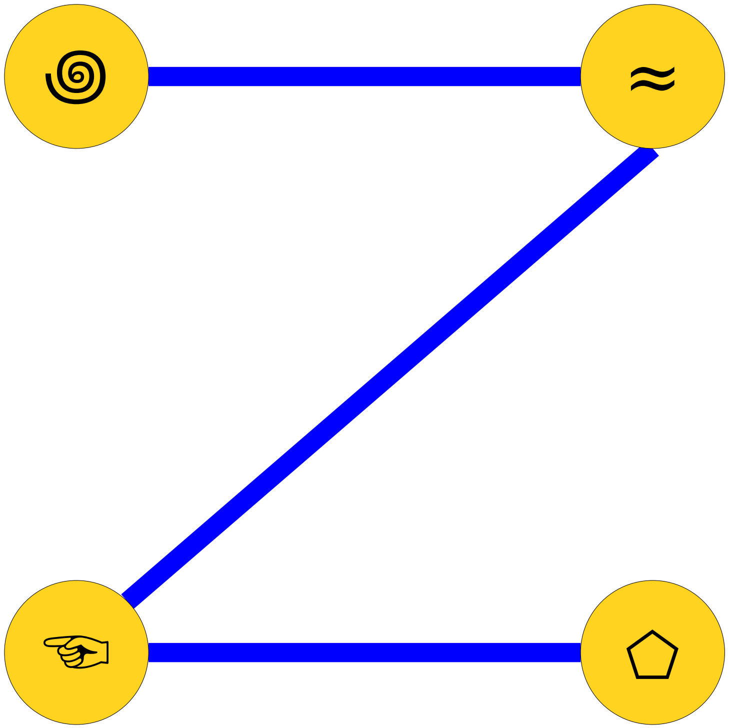
Now, consider the walk traced out by the train from the starting segment. That segment has indegree zero because it does not begin with a teleporter, so by our previous theorem this walk is a path. There are only finitely many segments and our path never revisits one, so eventually the path ends at a node with outdegree zero. The only node with this property is the end segment, so the train eventually reaches the end of the track. ■

Graph Complements

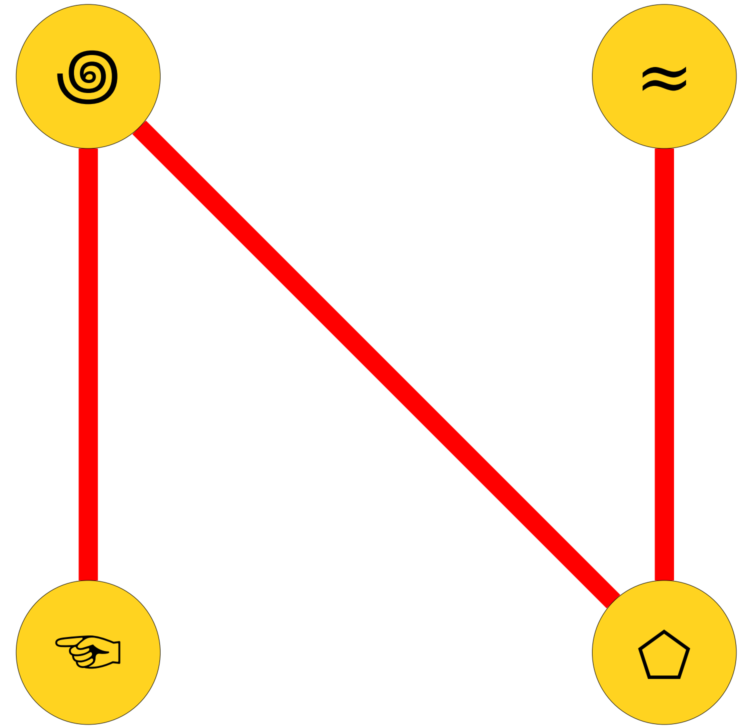






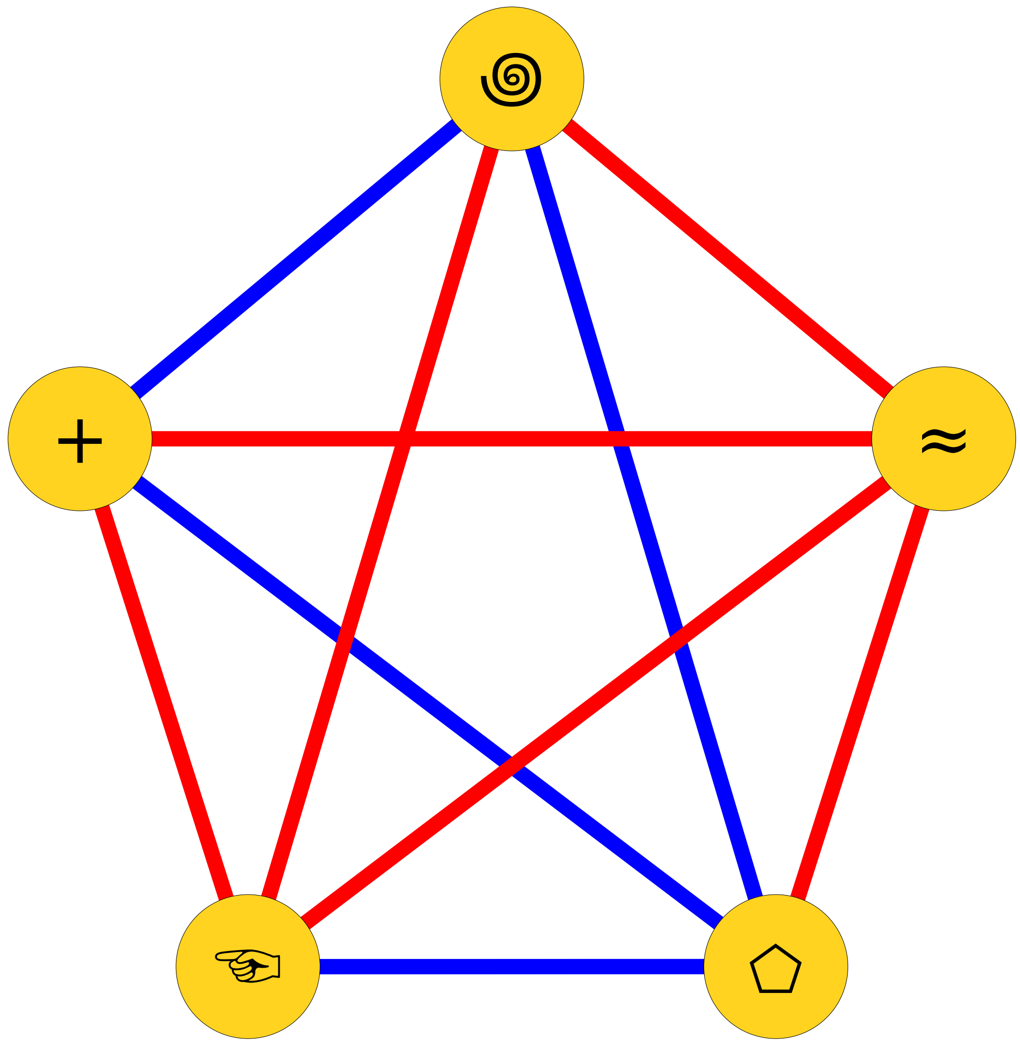


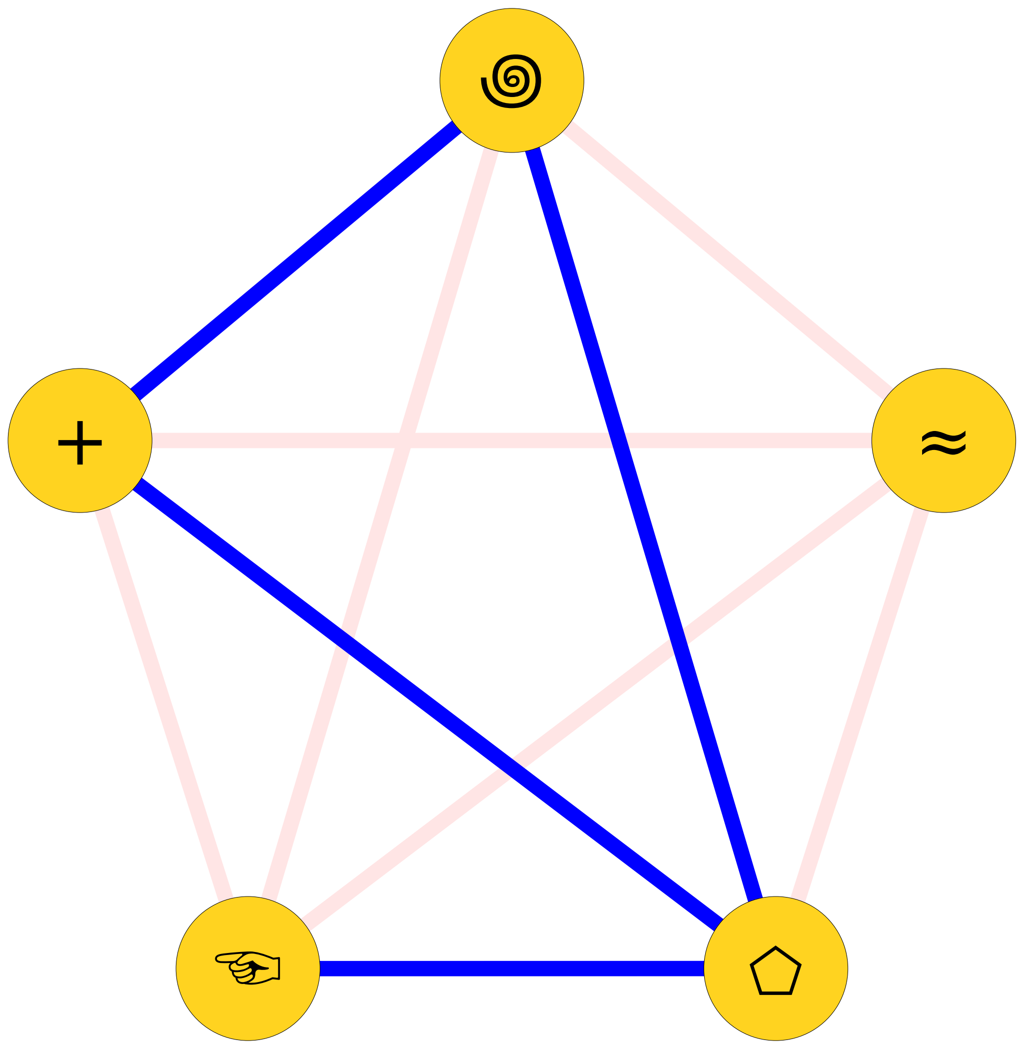
Graph G



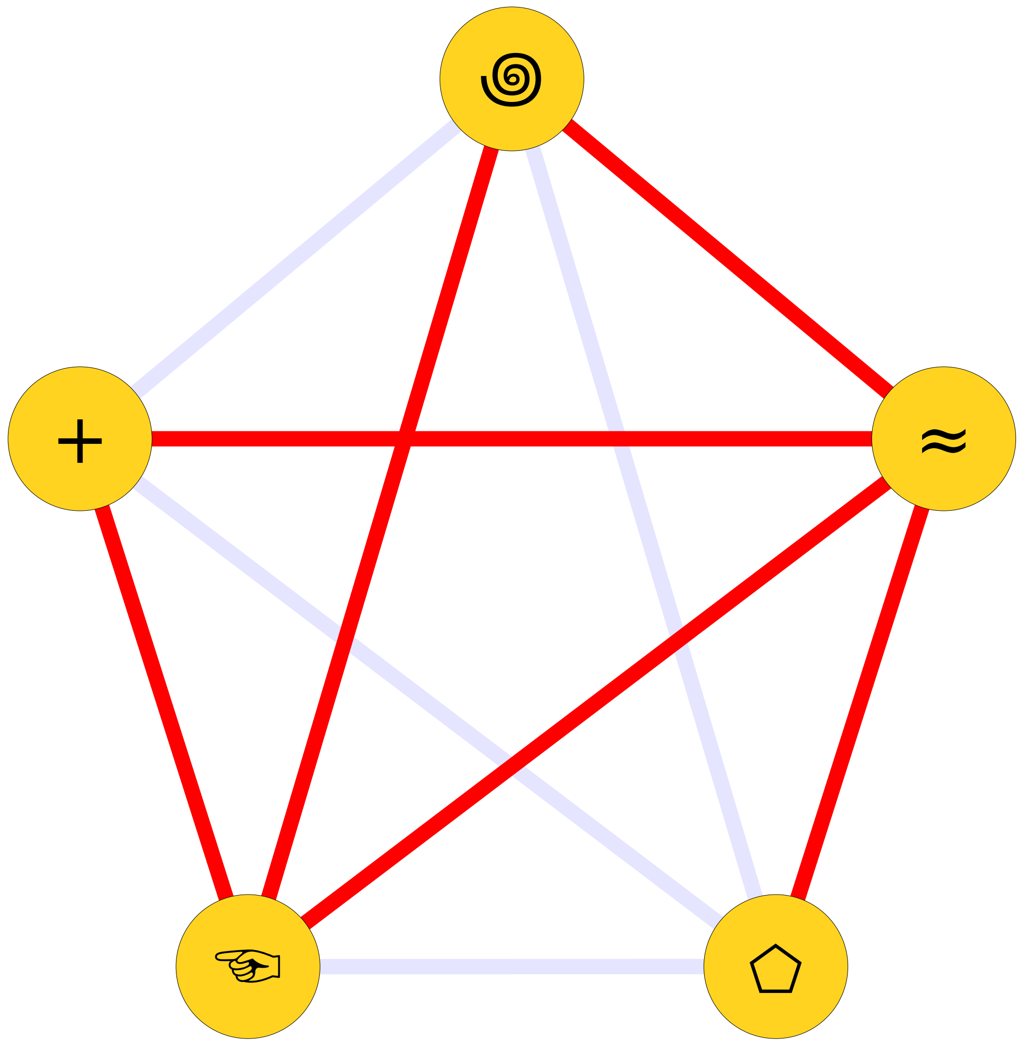
Graph G^c

Let $G = (V, E)$ be an undirected graph.
The ***complement of G*** is the graph $G^c = (V, E^c)$, where
$E^c = \{ \{u, v\} \mid u \in V, v \in V, u \neq v, \text{ and } \{u, v\} \notin E \}$

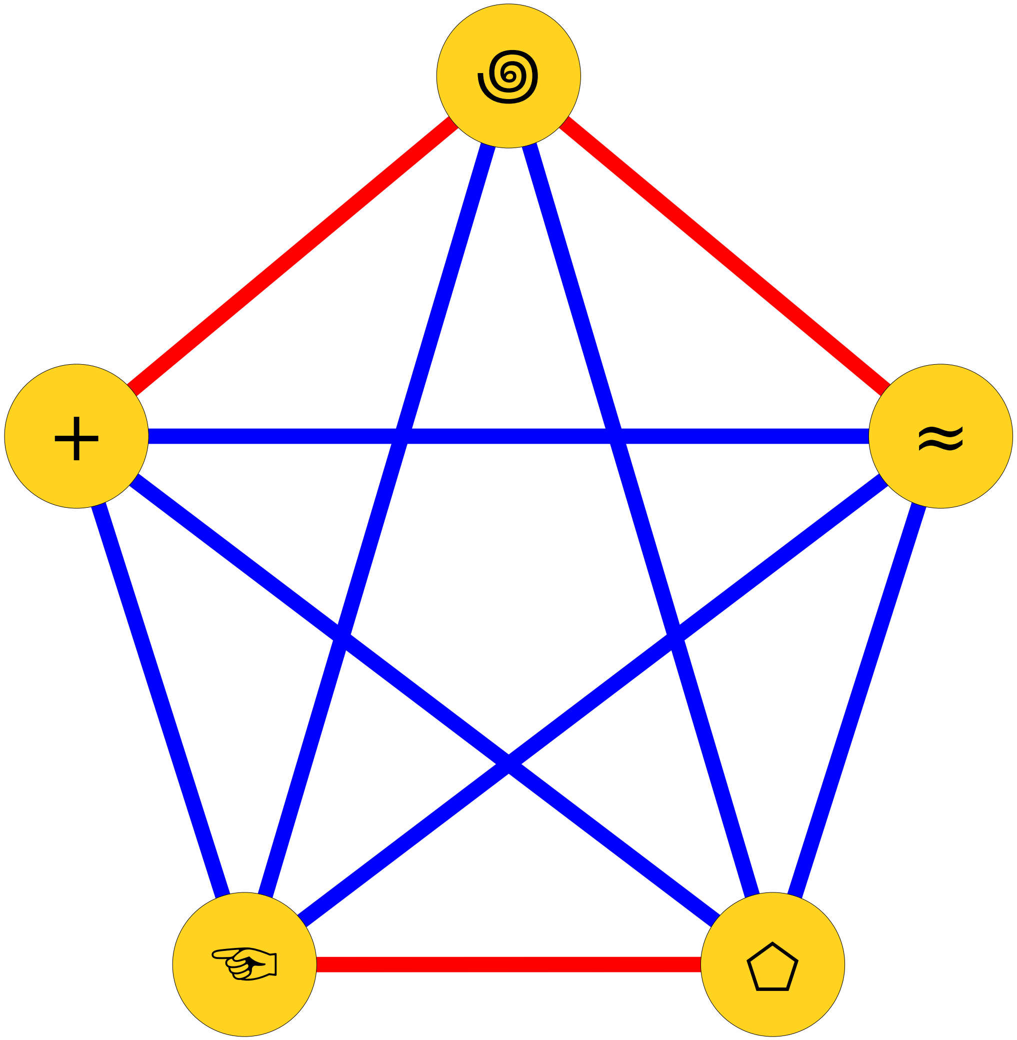


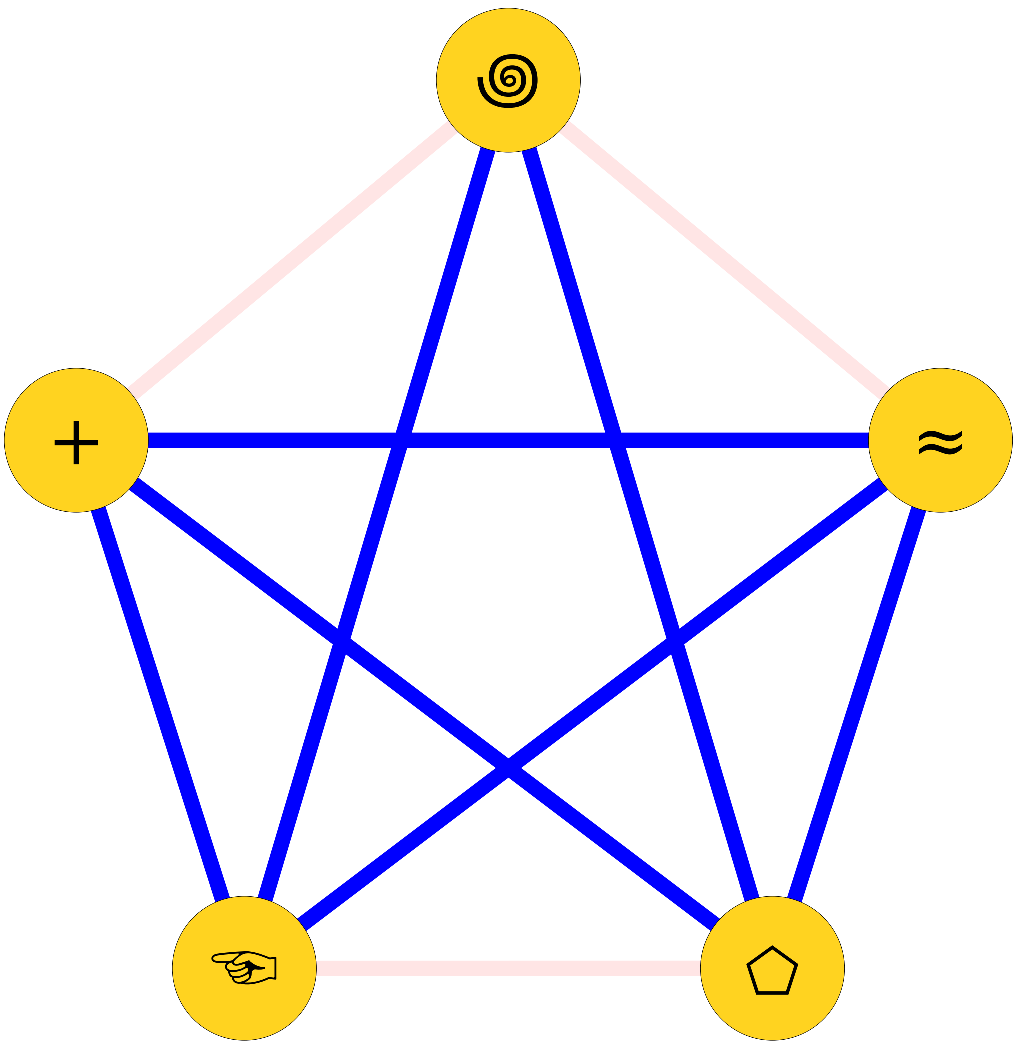


Graph G isn't connected.

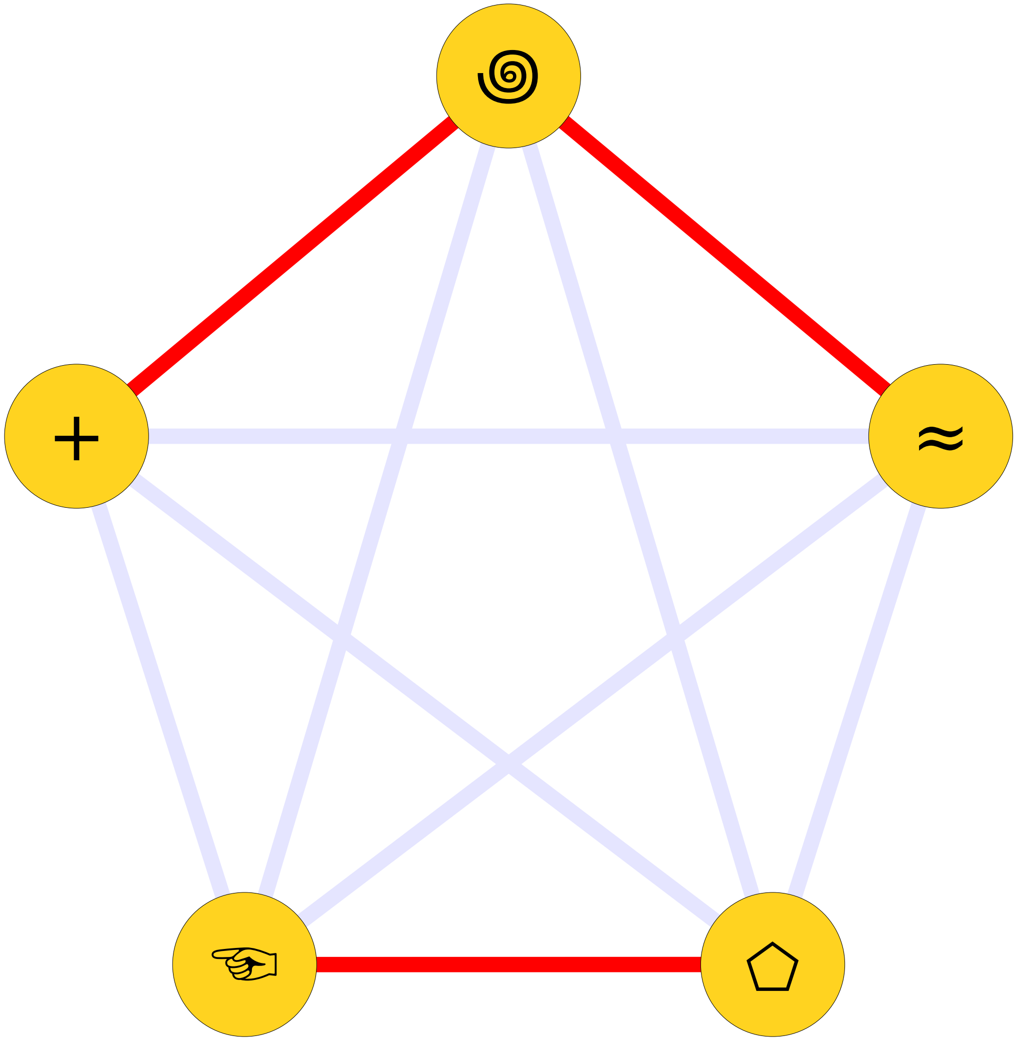


Graph G^c is connected.





Graph G is connected.



Graph G^c isn't connected.

Theorem: For any graph $G = (V, E)$, at least one of G and G^c is connected.

Proving a Disjunction

- We need to prove the statement

  G is connected $\lor$ G^c is connected.

- Here's a neat observation.
  - If G is connected, we're done.
  - Otherwise, G isn't connected, and we have to prove that G^c is connected.
- We will therefore prove

  G is not connected $\to$ G^c is connected.

For any graph $G = (V, E)$,
at least one of G and G^c is connected.

Proving a Disjunction

- We need to prove the statement

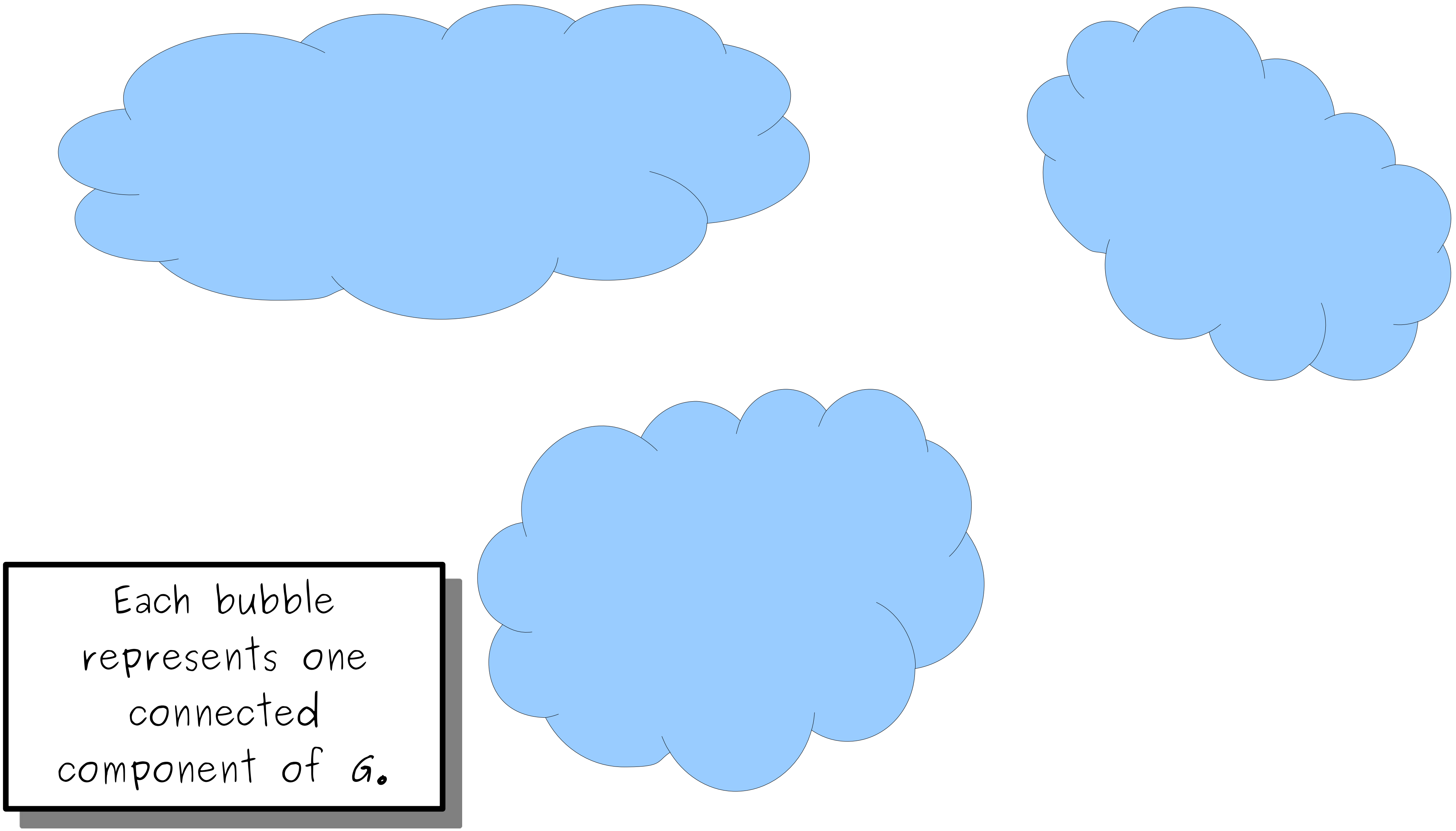
G is connected $\lor$ ***G^c is connected.***

- Here's a neat observation.
  - If G is connected, we're done.
  - Otherwise, G isn't connected, and we have to prove that G^c is connected.
- We will therefore prove

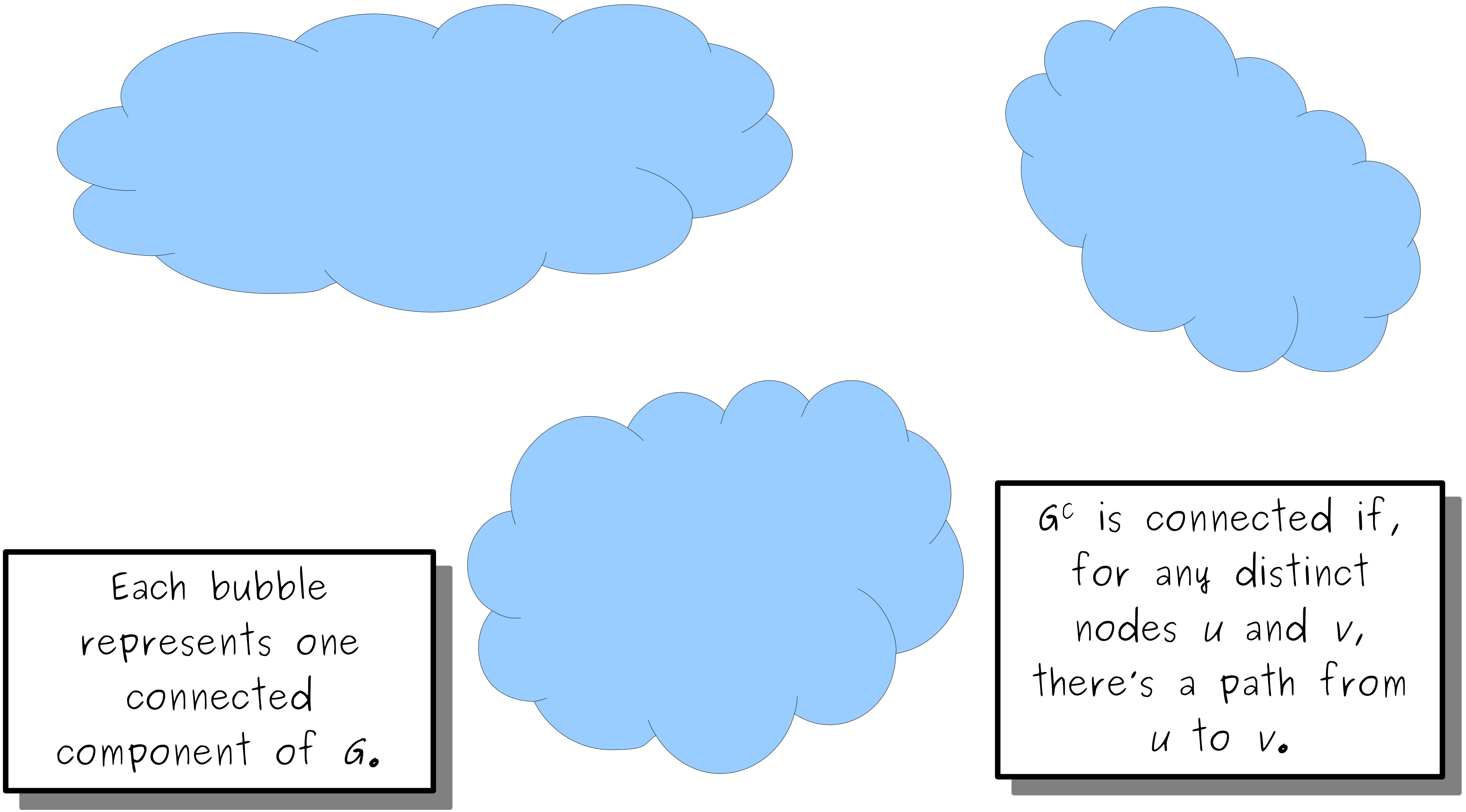
G is not connected $\to$ ***G^c is connected.***

For any graph $G = (V, E)$,
if G is not connected, then G^c is connected.

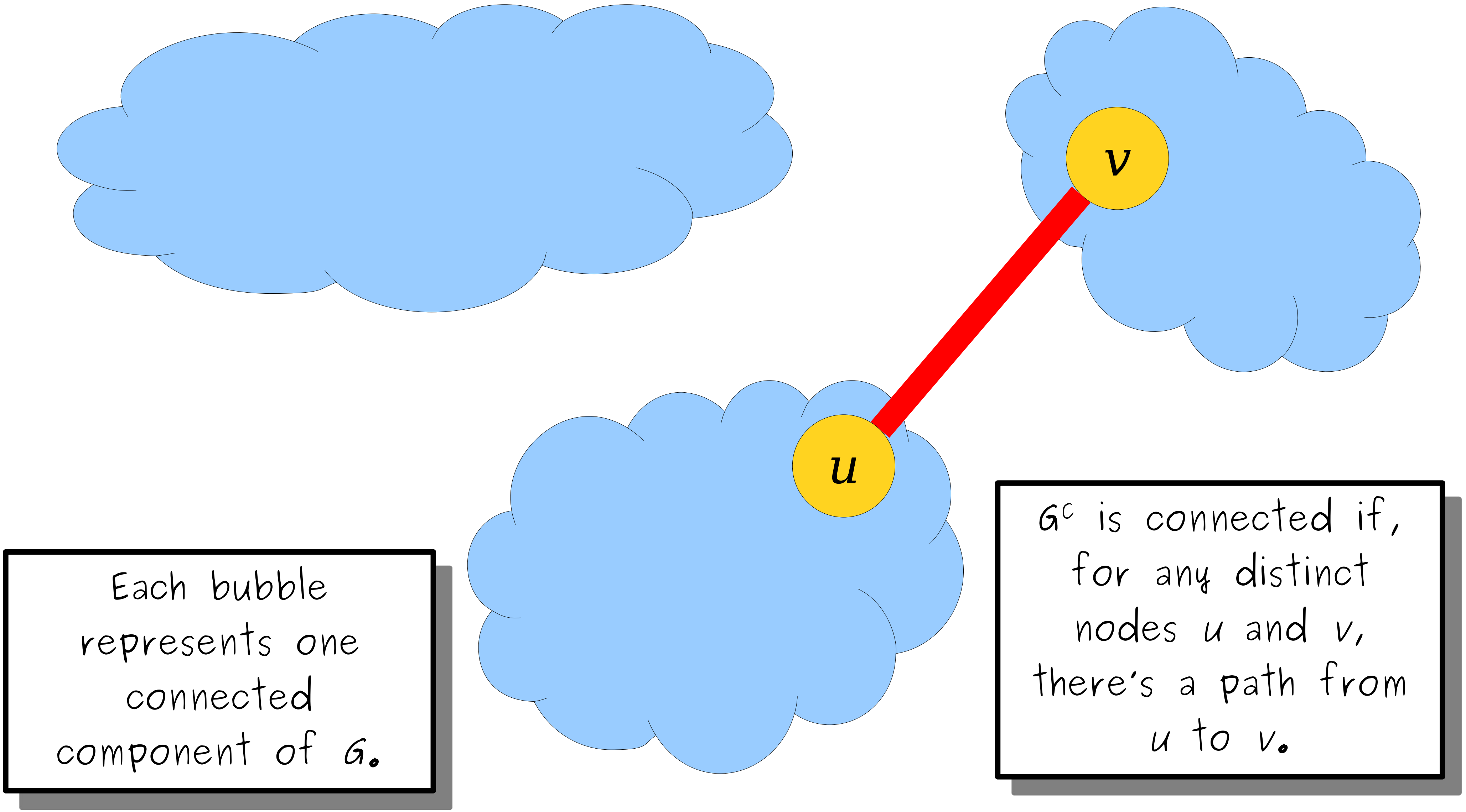
For any graph $G = (V, E)$,
if G is not connected, then G^c is connected.



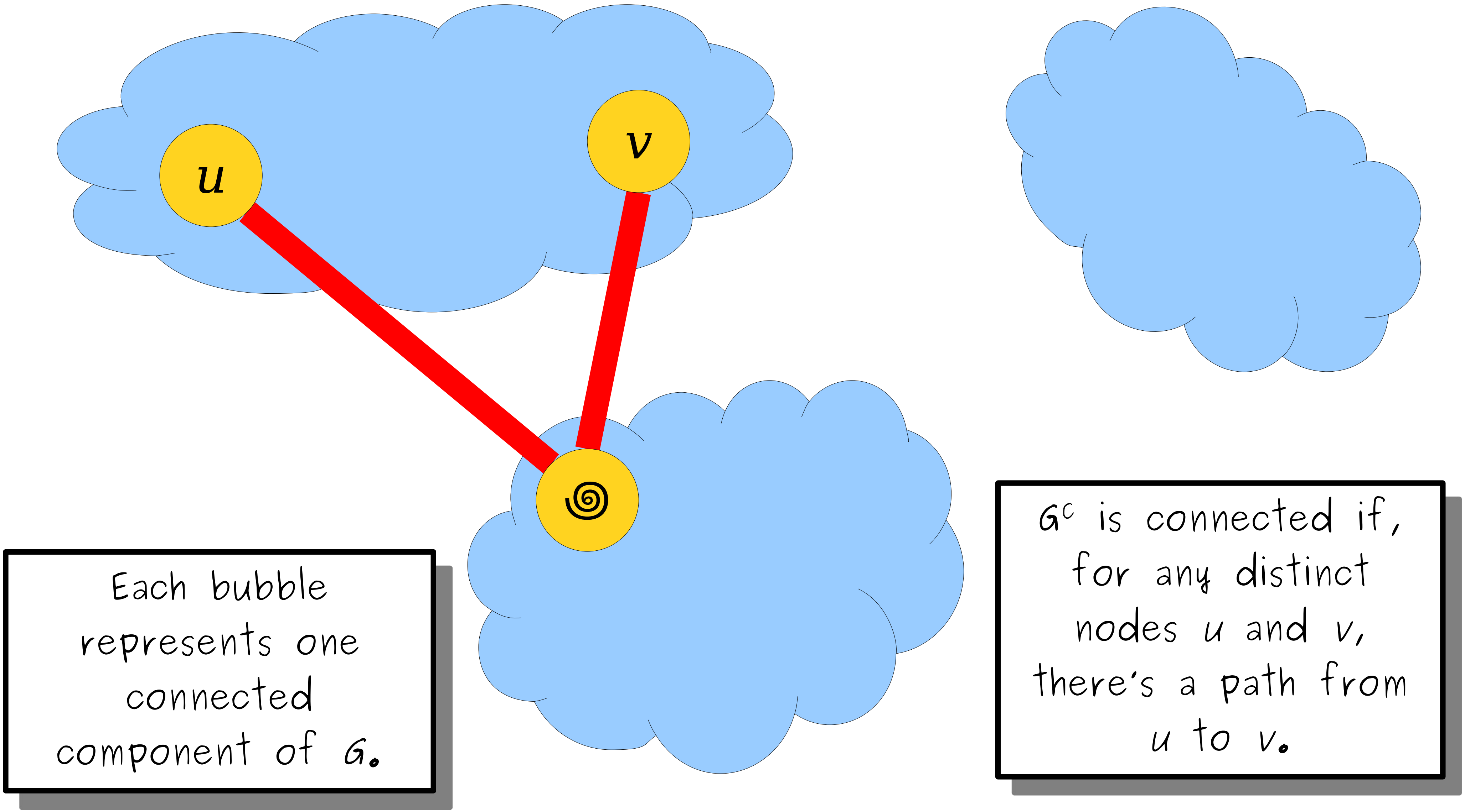
For any graph $G = (V, E)$,
if G is not connected, then G^c is connected.



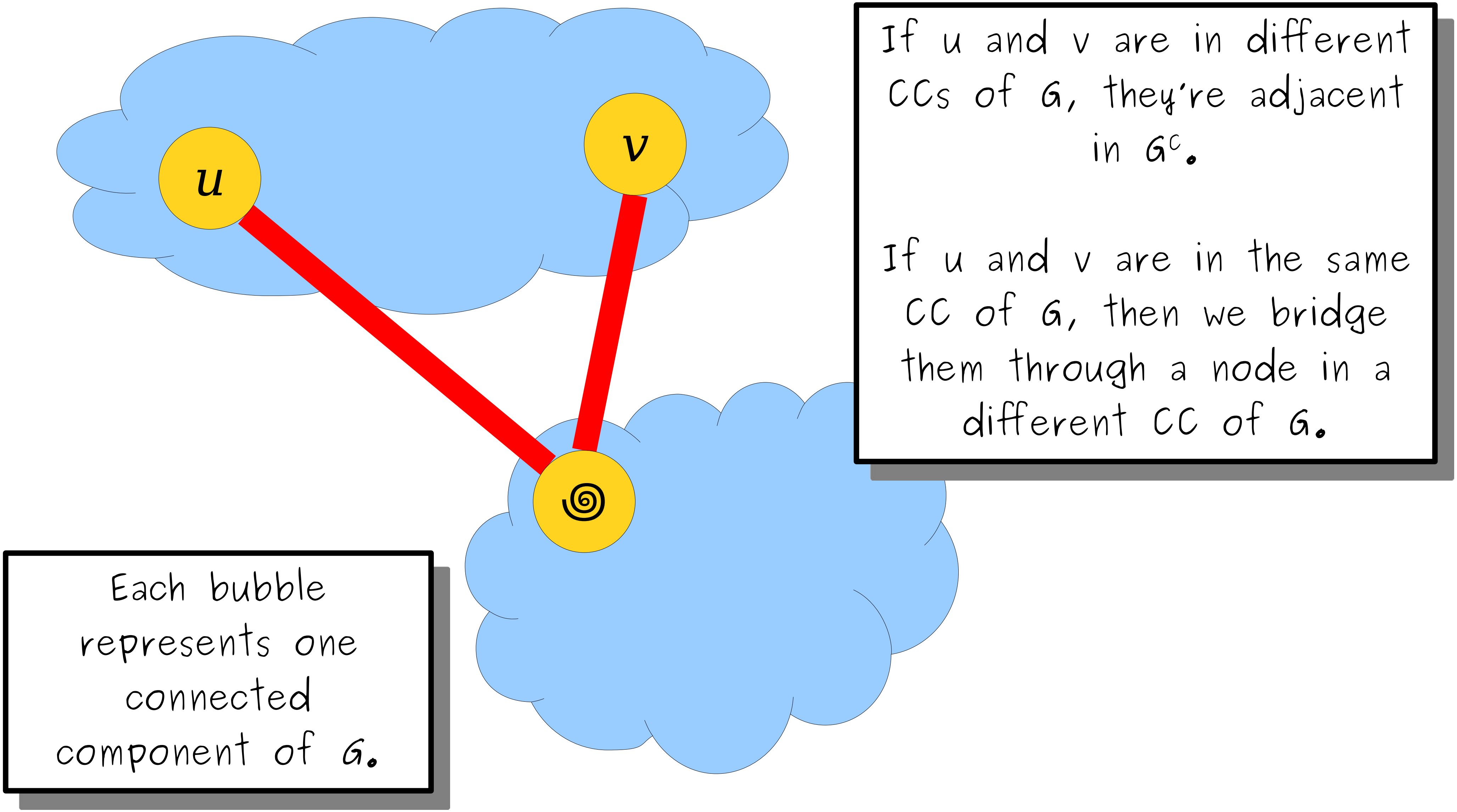
For any graph $G = (V, E)$,
if G is not connected, **then G^c is connected**.



For any graph $G = (V, E)$,
if G is not connected, then G^c is connected.



For any graph $G = (V, E)$,
if G is not connected, then G^c is connected.



For any graph $G = (V, E)$,
if G is not connected, **then G^c is connected**.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof:

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$. We need to show that there is a path from u to v in G^c.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$. We need to show that there is a path from u to v in G^c. We consider two cases:

Case 1: u and v are in different connected components of G.

Case 2: u and v are in the same connected component of G.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$. We need to show that there is a path from u to v in G^c. We consider two cases:

Case 1: u and v are in different connected components of G. This means that $\{u, v\} \notin E$, since otherwise the path u, v would make u reachable from v and they'd be in the same connected component of G.

Case 2: u and v are in the same connected component of G.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$. We need to show that there is a path from u to v in G^c. We consider two cases:

Case 1: u and v are in different connected components of G. This means that $\{u, v\} \notin E$, since otherwise the path u, v would make u reachable from v and they'd be in the same connected component of G. Therefore, we see that $\{u, v\} \in E^c$, and so there is a path (namely, u, v) from u to v in G^c.

Case 2: u and v are in the same connected component of G.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$. We need to show that there is a path from u to v in G^c. We consider two cases:

Case 1: u and v are in different connected components of G. This means that $\{u, v\} \notin E$, since otherwise the path u, v would make u reachable from v and they'd be in the same connected component of G. Therefore, we see that $\{u, v\} \in E^c$, and so there is a path (namely, u, v) from u to v in G^c.

Case 2: u and v are in the same connected component of G. Since G is not connected, there are at least two connected components of G.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$. We need to show that there is a path from u to v in G^c. We consider two cases:

Case 1: u and v are in different connected components of G. This means that $\{u, v\} \notin E$, since otherwise the path u, v would make u reachable from v and they'd be in the same connected component of G. Therefore, we see that $\{u, v\} \in E^c$, and so there is a path (namely, u, v) from u to v in G^c.

Case 2: u and v are in the same connected component of G. Since G is not connected, there are at least two connected components of G. Pick any node z that belongs to a different connected component of G than u and v.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$. We need to show that there is a path from u to v in G^c. We consider two cases:

Case 1: u and v are in different connected components of G. This means that $\{u, v\} \notin E$, since otherwise the path u, v would make u reachable from v and they'd be in the same connected component of G. Therefore, we see that $\{u, v\} \in E^c$, and so there is a path (namely, u, v) from u to v in G^c.

Case 2: u and v are in the same connected component of G. Since G is not connected, there are at least two connected components of G. Pick any node z that belongs to a different connected component of G than u and v. Then by the reasoning from Case 1 we know that $\{u, z\} \in E^c$ and $\{z, v\} \in E^c$.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$. We need to show that there is a path from u to v in G^c. We consider two cases:

Case 1: u and v are in different connected components of G. This means that $\{u, v\} \notin E$, since otherwise the path u, v would make u reachable from v and they'd be in the same connected component of G. Therefore, we see that $\{u, v\} \in E^c$, and so there is a path (namely, u, v) from u to v in G^c.

Case 2: u and v are in the same connected component of G. Since G is not connected, there are at least two connected components of G. Pick any node z that belongs to a different connected component of G than u and v. Then by the reasoning from Case 1 we know that $\{u, z\} \in E^c$ and $\{z, v\} \in E^c$. This gives a path u, z, v in G^c from u to v.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$. We need to show that there is a path from u to v in G^c. We consider two cases:

Case 1: u and v are in different connected components of G. This means that $\{u, v\} \notin E$, since otherwise the path u, v would make u reachable from v and they'd be in the same connected component of G. Therefore, we see that $\{u, v\} \in E^c$, and so there is a path (namely, u, v) from u to v in G^c.

Case 2: u and v are in the same connected component of G. Since G is not connected, there are at least two connected components of G. Pick any node z that belongs to a different connected component of G than u and v. Then by the reasoning from Case 1 we know that $\{u, z\} \in E^c$ and $\{z, v\} \in E^c$. This gives a path u, z, v in G^c from u to v.

In either case, we find a path from u to v in G^c, as required.

Theorem: If $G = (V, E)$ is a graph, then at least one of G and G^c is connected.

Proof: Let $G = (V, E)$ be an arbitrary graph and assume G is not connected. We need to show that $G^c = (V, E^c)$ is connected. To do so, consider any two distinct nodes $u, v \in V$. We need to show that there is a path from u to v in G^c. We consider two cases:

Case 1: u and v are in different connected components of G. This means that $\{u, v\} \notin E$, since otherwise the path u, v would make u reachable from v and they'd be in the same connected component of G. Therefore, we see that $\{u, v\} \in E^c$, and so there is a path (namely, u, v) from u to v in G^c.

Case 2: u and v are in the same connected component of G. Since G is not connected, there are at least two connected components of G. Pick any node z that belongs to a different connected component of G than u and v. Then by the reasoning from Case 1 we know that $\{u, z\} \in E^c$ and $\{z, v\} \in E^c$. This gives a path u, z, v in G^c from u to v.

In either case, we find a path from u to v in G^c, as required. ■

Recap for Today

- We can use ***walks*** and ***closed walks*** to travel around a graph. Walks and closed walks that don't repeat nodes or edges are called ***paths*** and ***cycles***, respectively.
- The ***indegree*** and ***outdegree*** of a node in a digraph are the number of edges entering or leaving the node, respectively.
- Digraphs where the indegree and outdegree of each node are at most one break apart into isolated paths and cycles.
- You can't trap a train on a track with teleporters, unless there's a teleporter behind the train.

Next Time

- ***The Pigeonhole Principle***
  - A simple, powerful, versatile theorem.
- ***Graph Theory Party Tricks***
  - Applying math to graphs of people!
- ***A Little Movie Puzzle***
  - Who watched what?