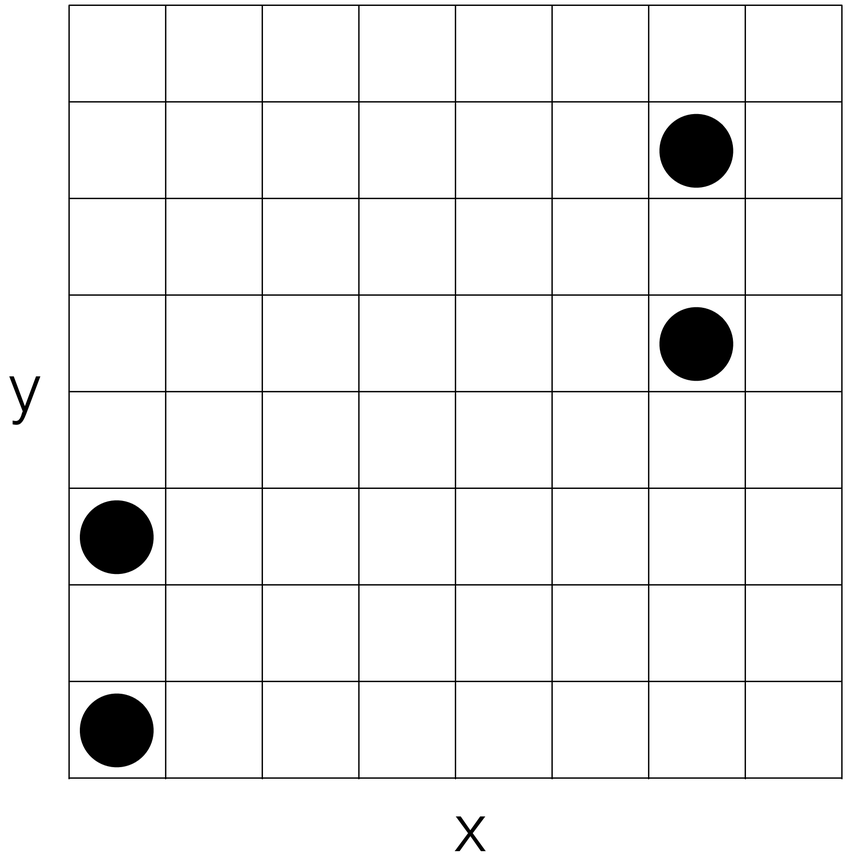
Hierarchical Data Cubes - a worked example
CSC544

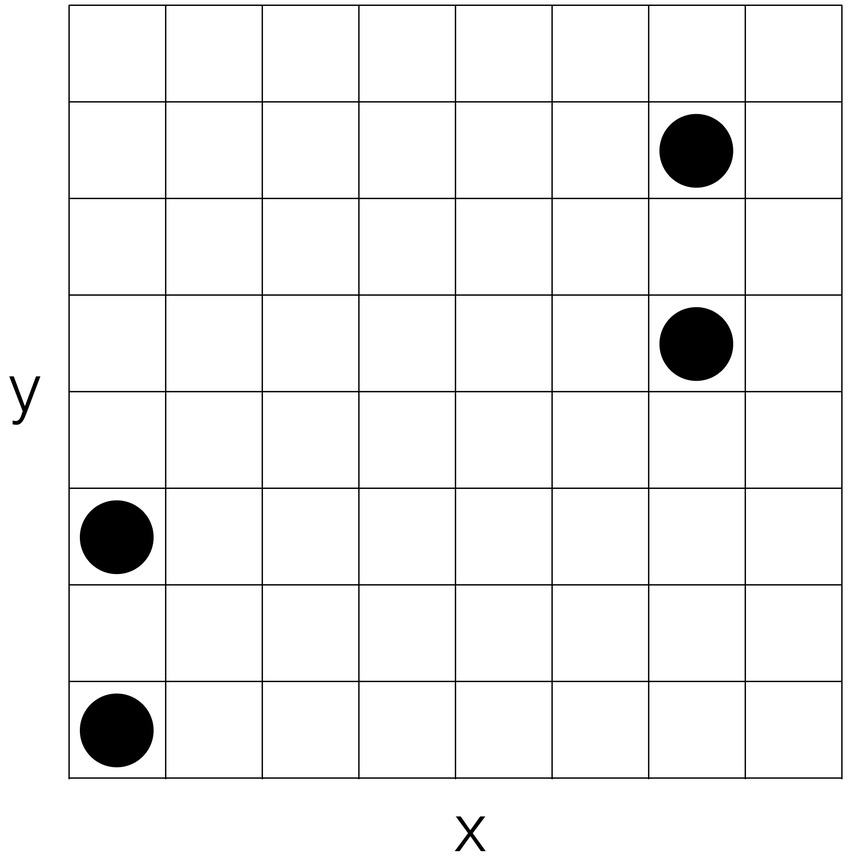
Let’s build a small 2D hierarchical cube

Two dimensions, 3 bits of resolution in each dimension

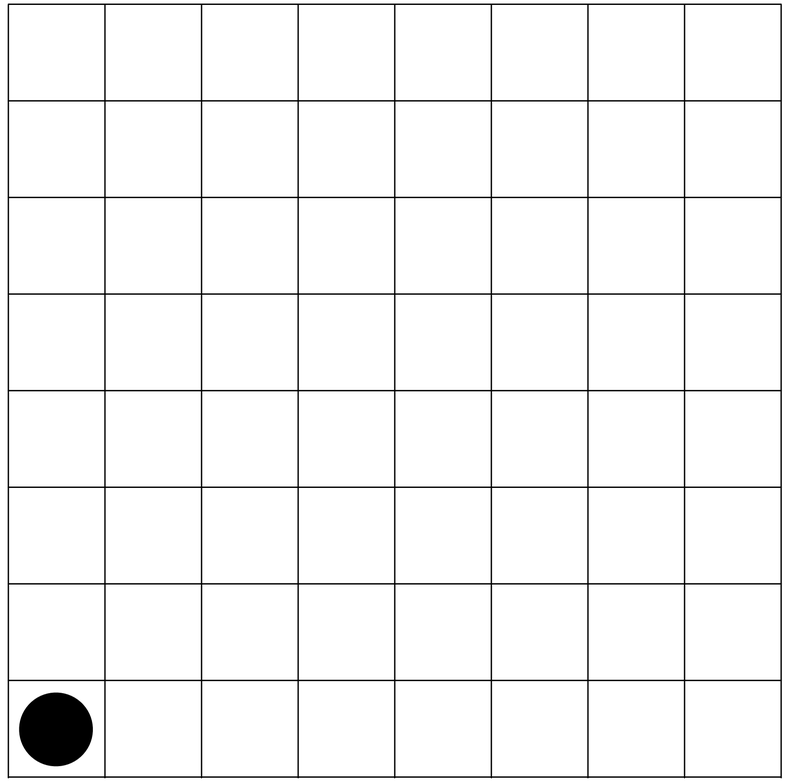


(0, 0)
(0, 2)
(6, 4)
(6, 6)

Two dimensions, 3 bits of resolution in each dimension



(0, 0)	**(000, 000)**
(0, 2)	**(000, 010)**
(6, 4)	**(110, 100)**
(6, 6)	**(110, 110)**

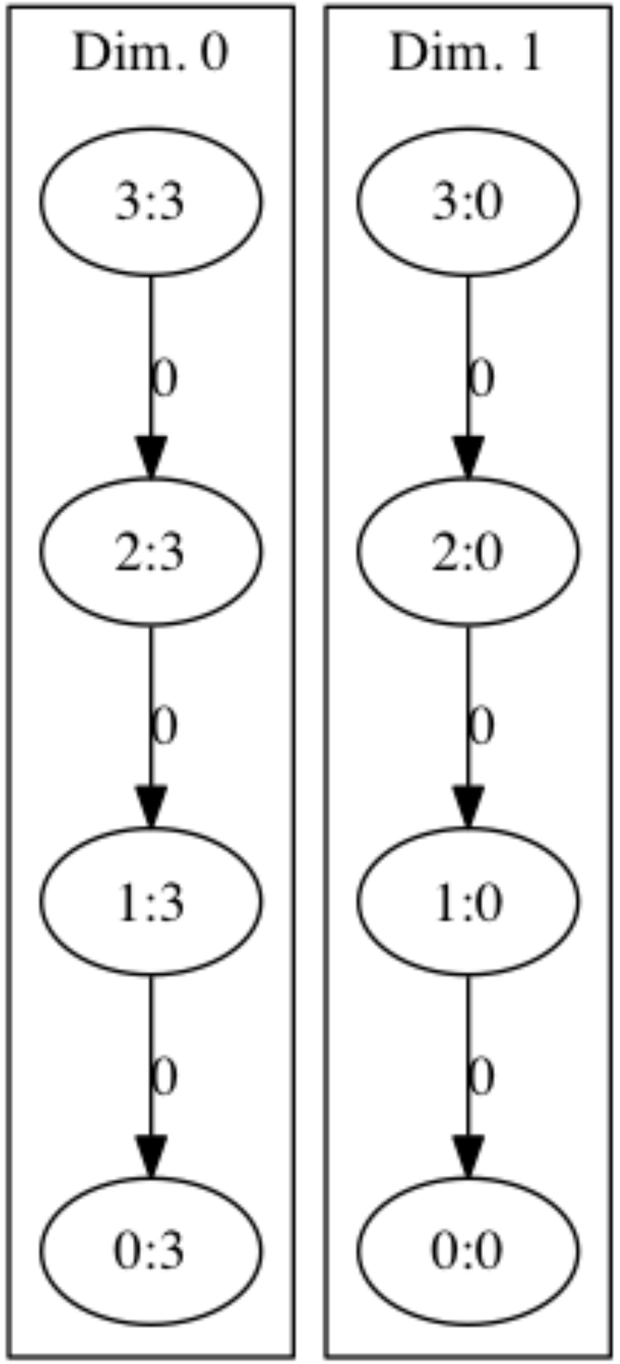


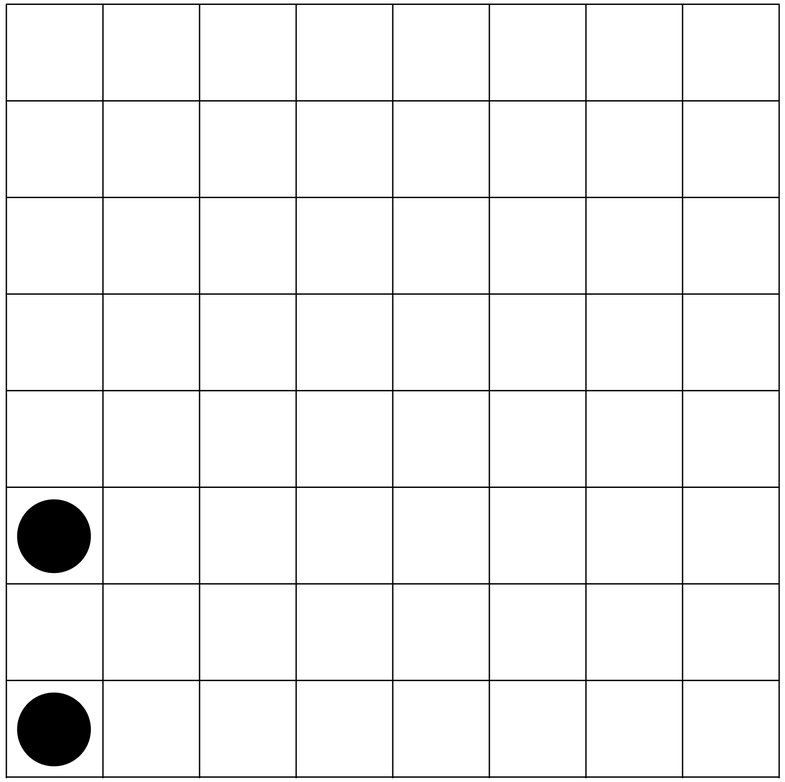
(000, 000)

(000, 010)

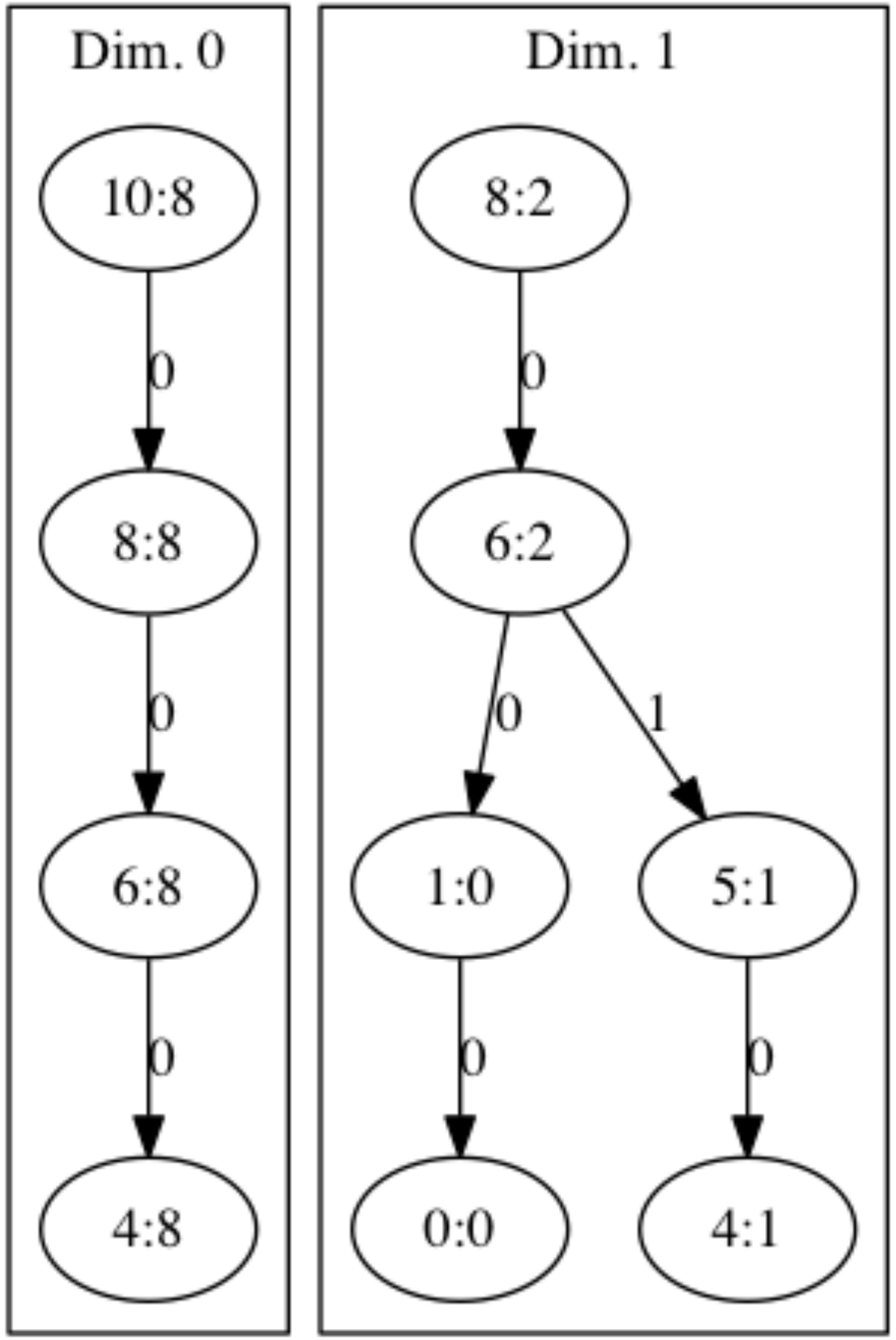
(110, 100)

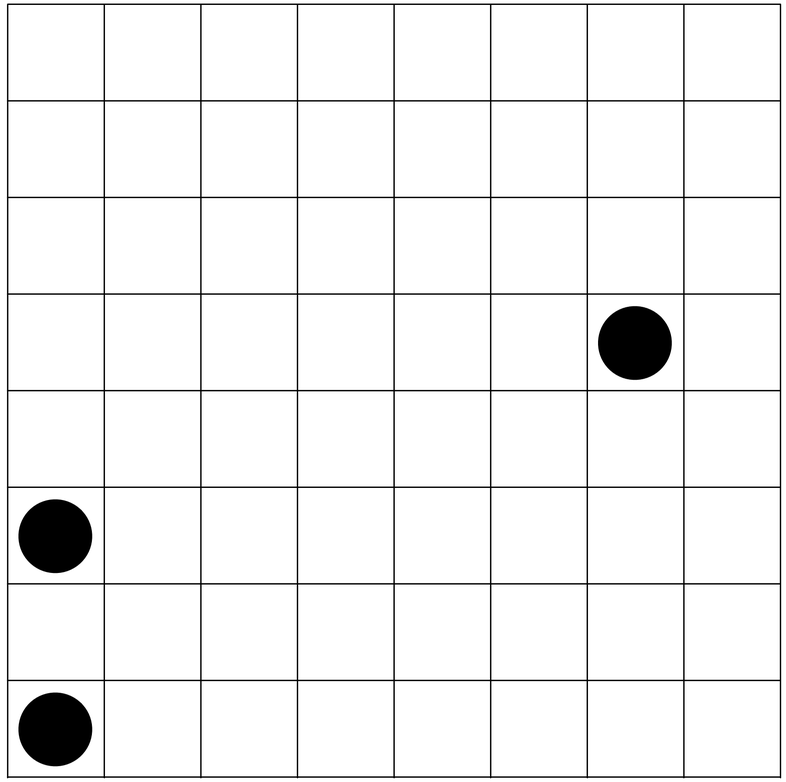
(110, 110)



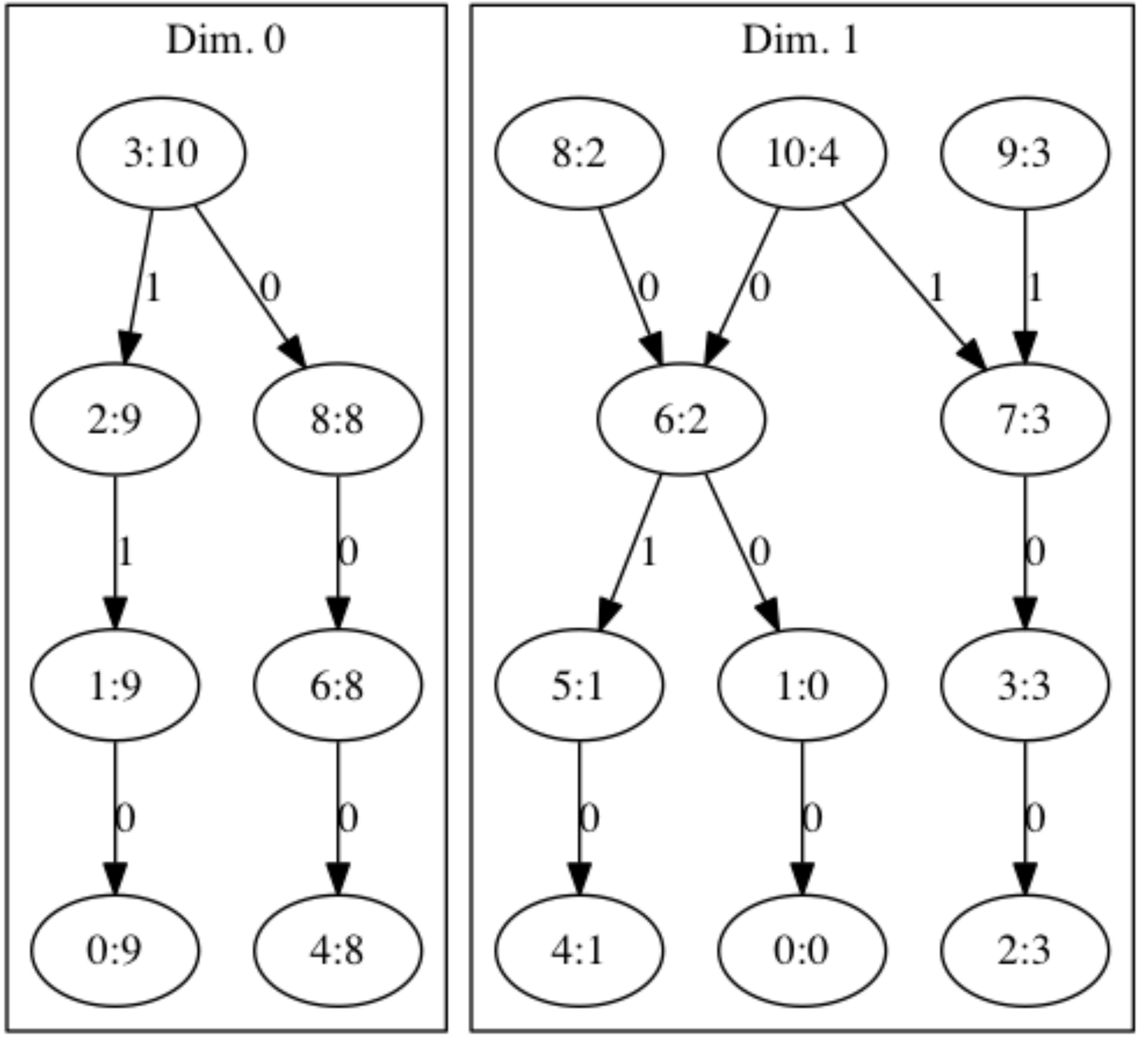


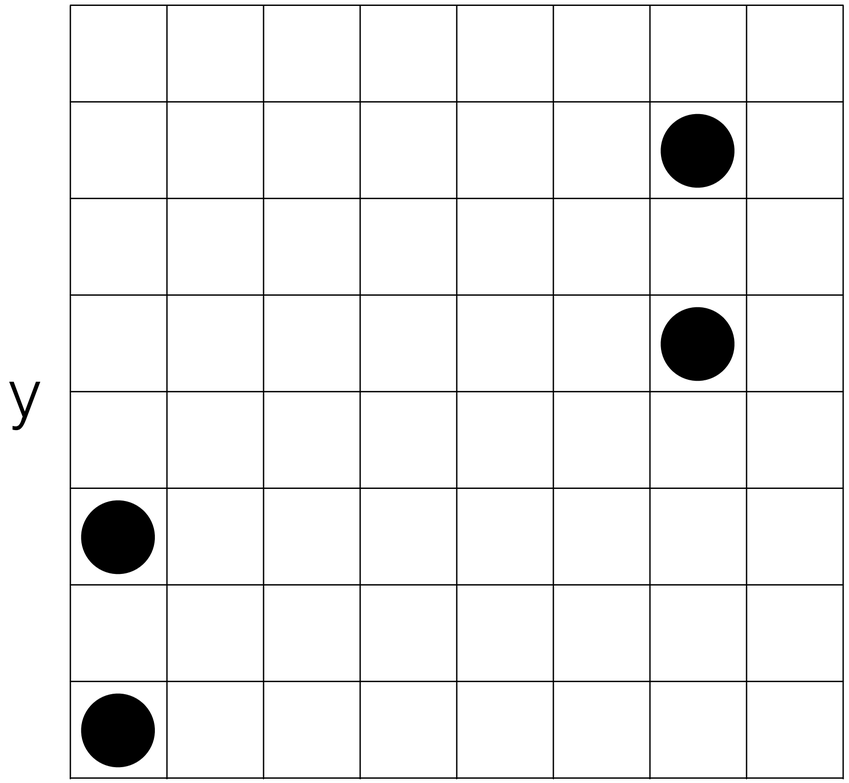
(000, 000)
(000, 010)
(110, 100)
(110, 110)



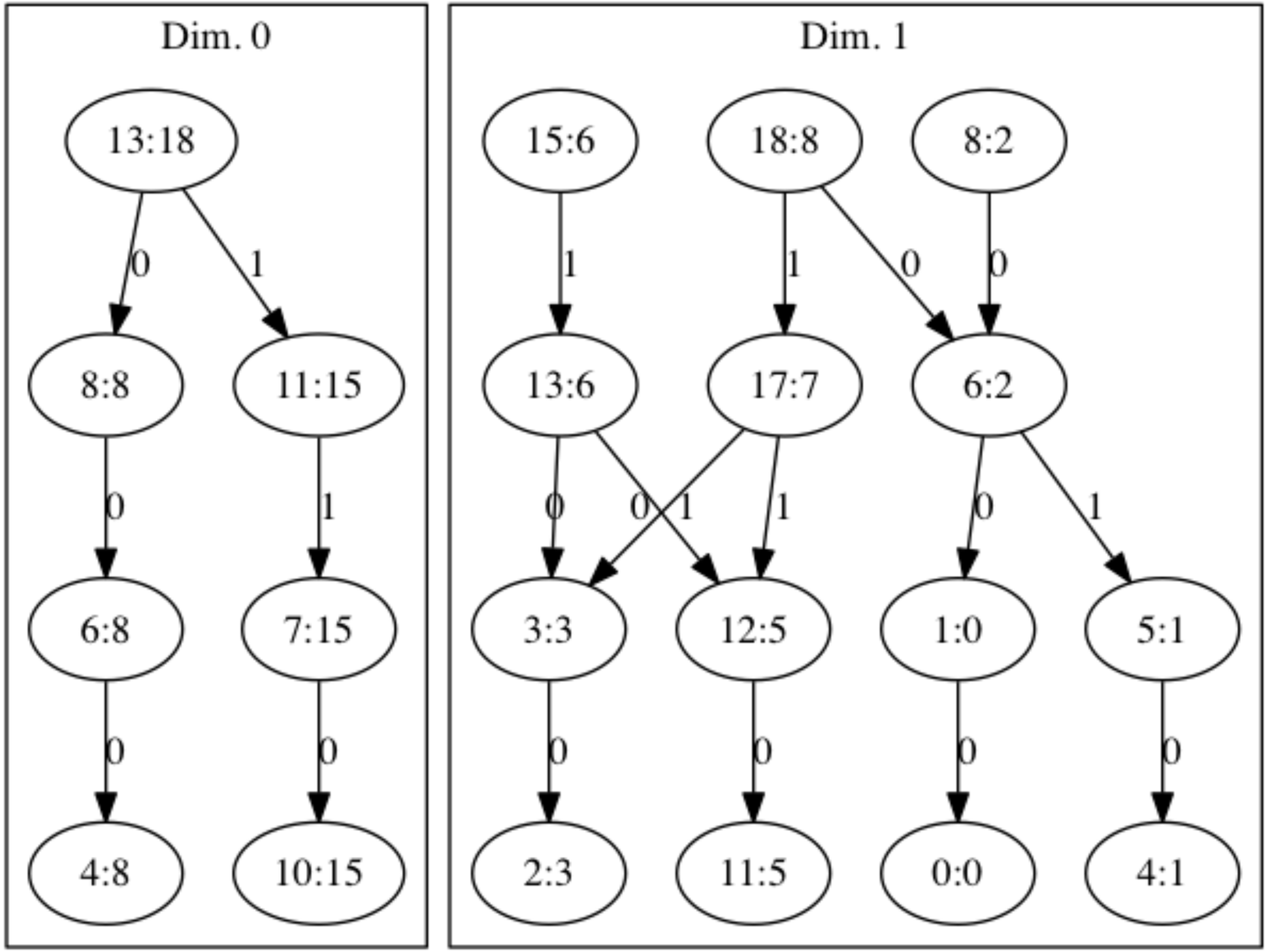


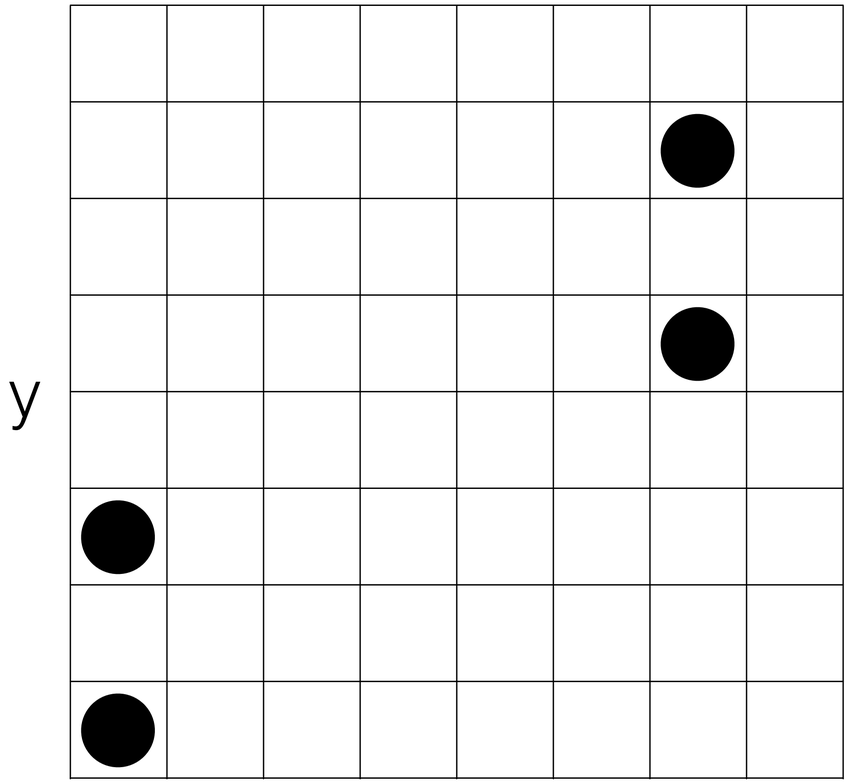
(000, 000)
(000, 010)
(110, 100)
(110, 110)



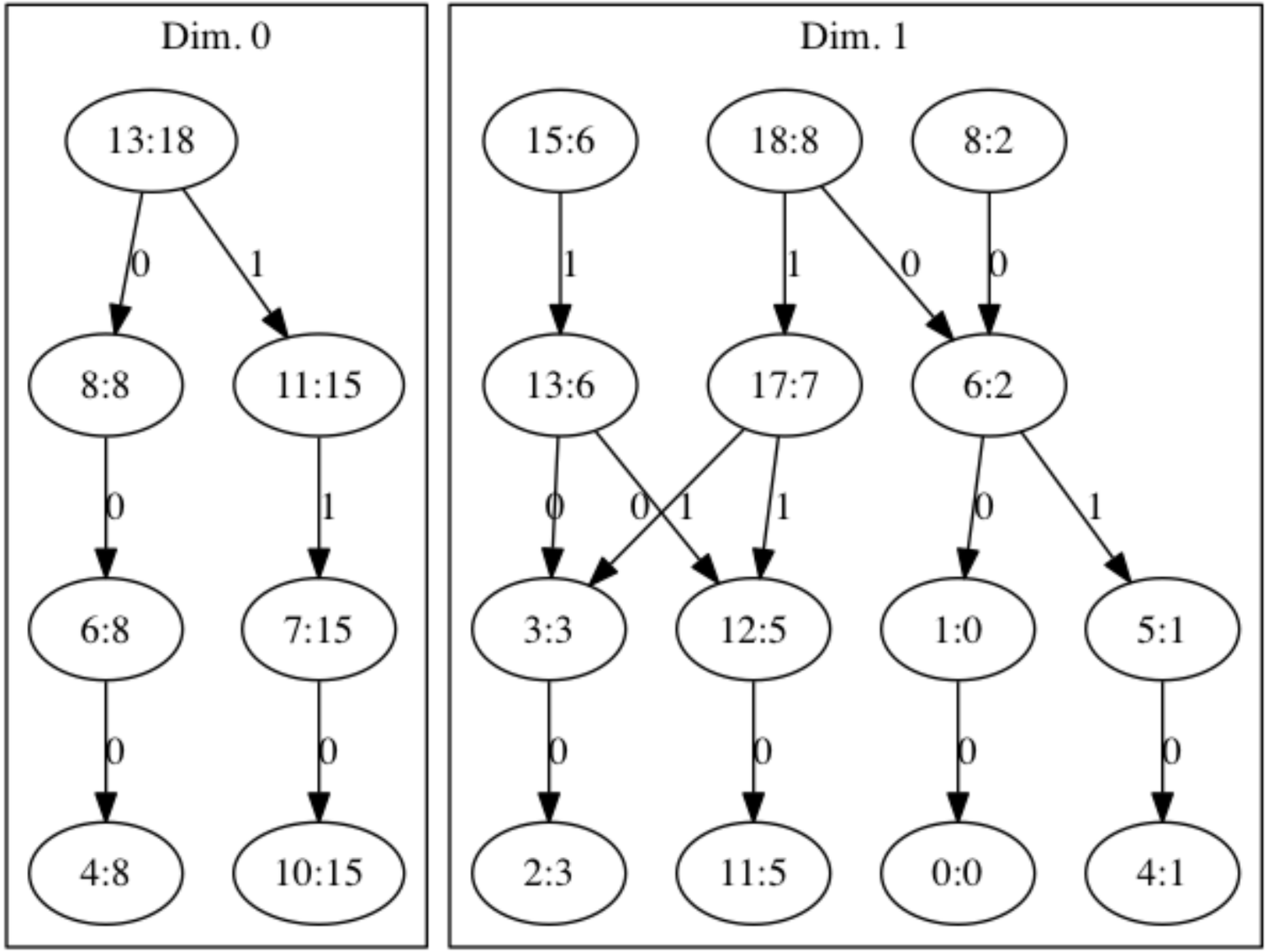


(000, 000)
(000, 010)
(110, 100)
(110, 110)



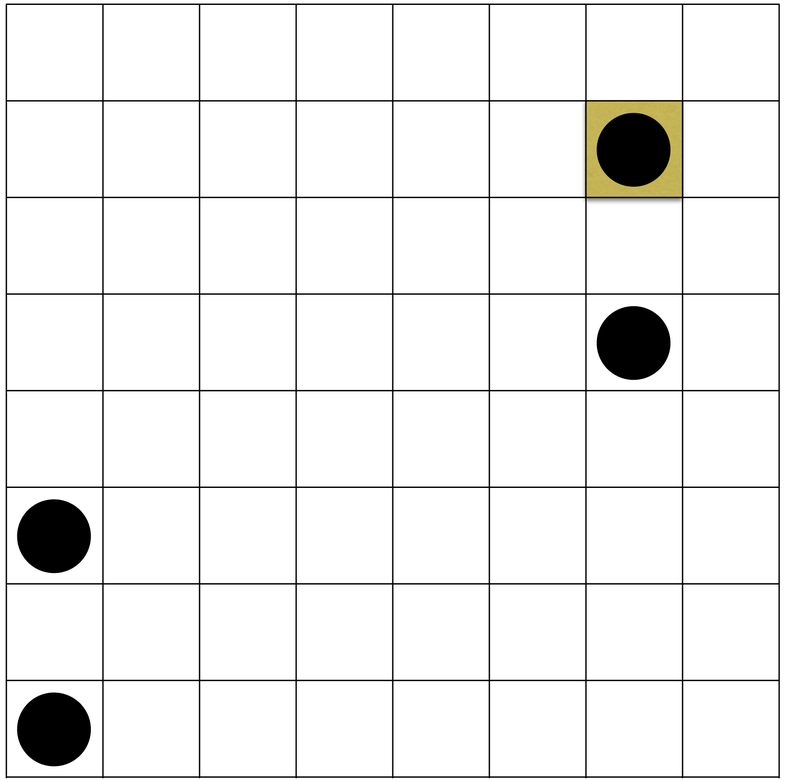


(000, 000)
(000, 010)
(110, 100)
(110, 110)

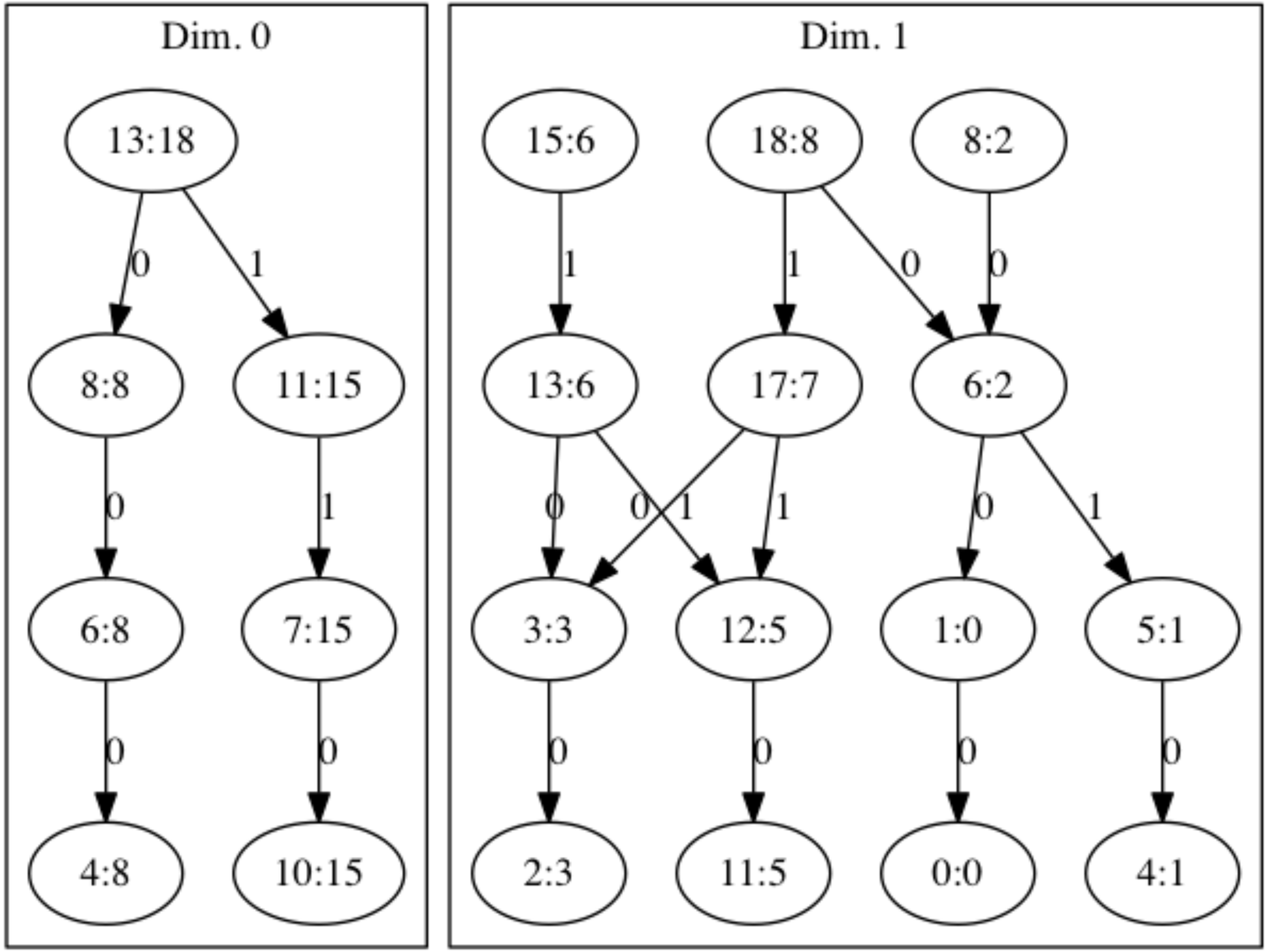


Let’s make some queries against it

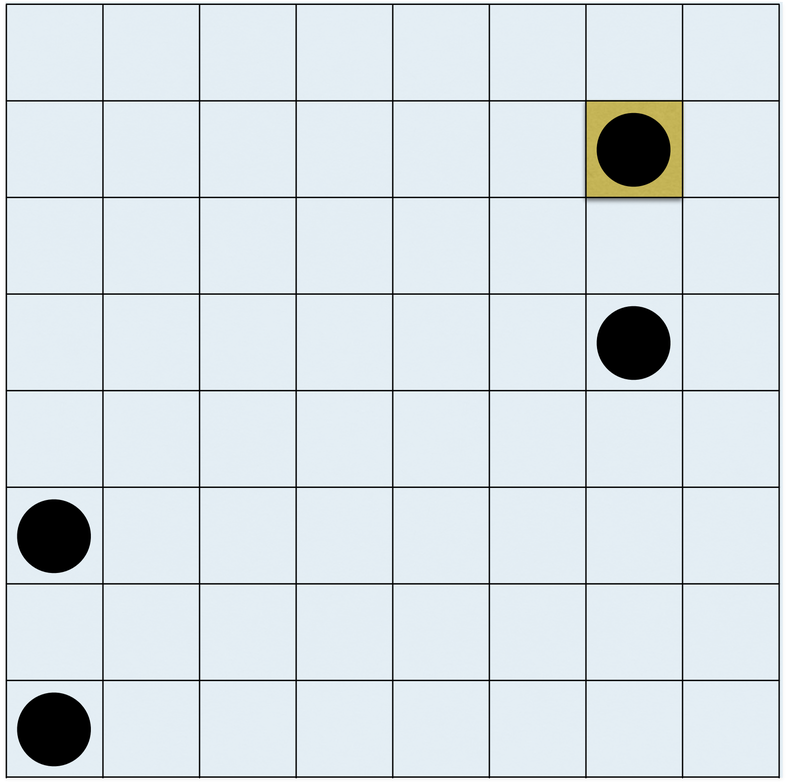
Example 1



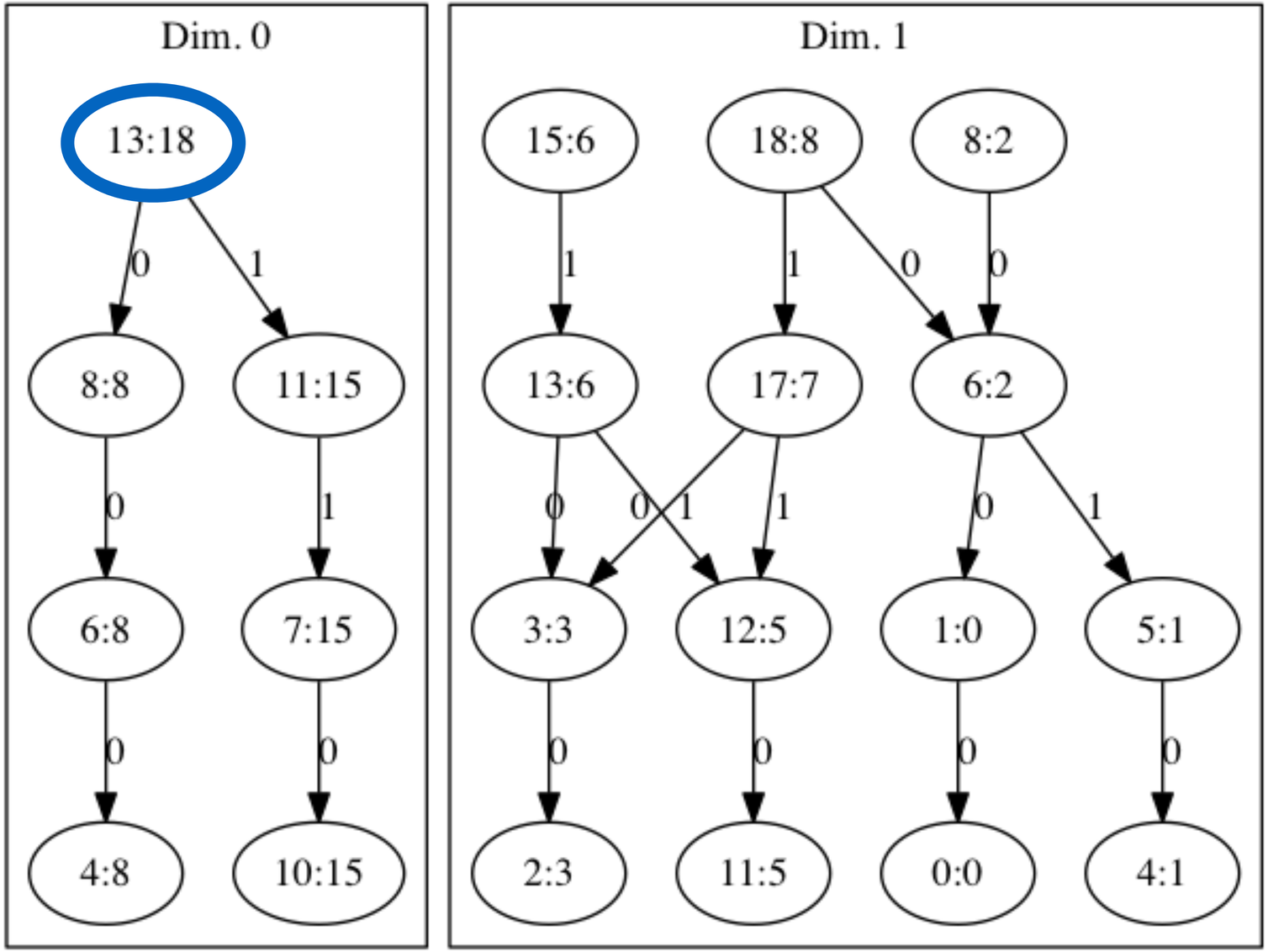
(000, 000)
(000, 010)
(110, 100)
(110, 110)



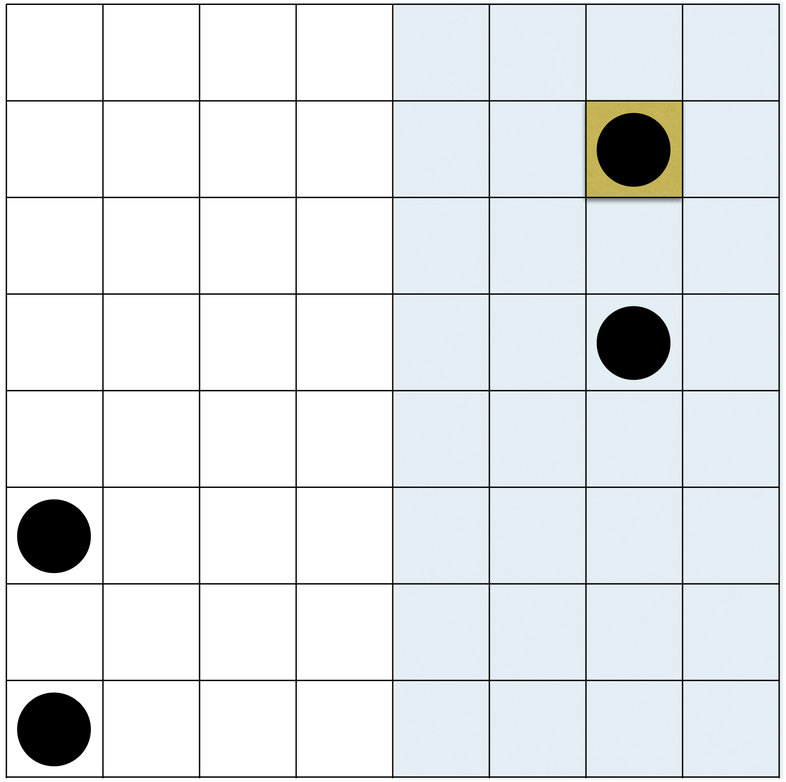
Example 1



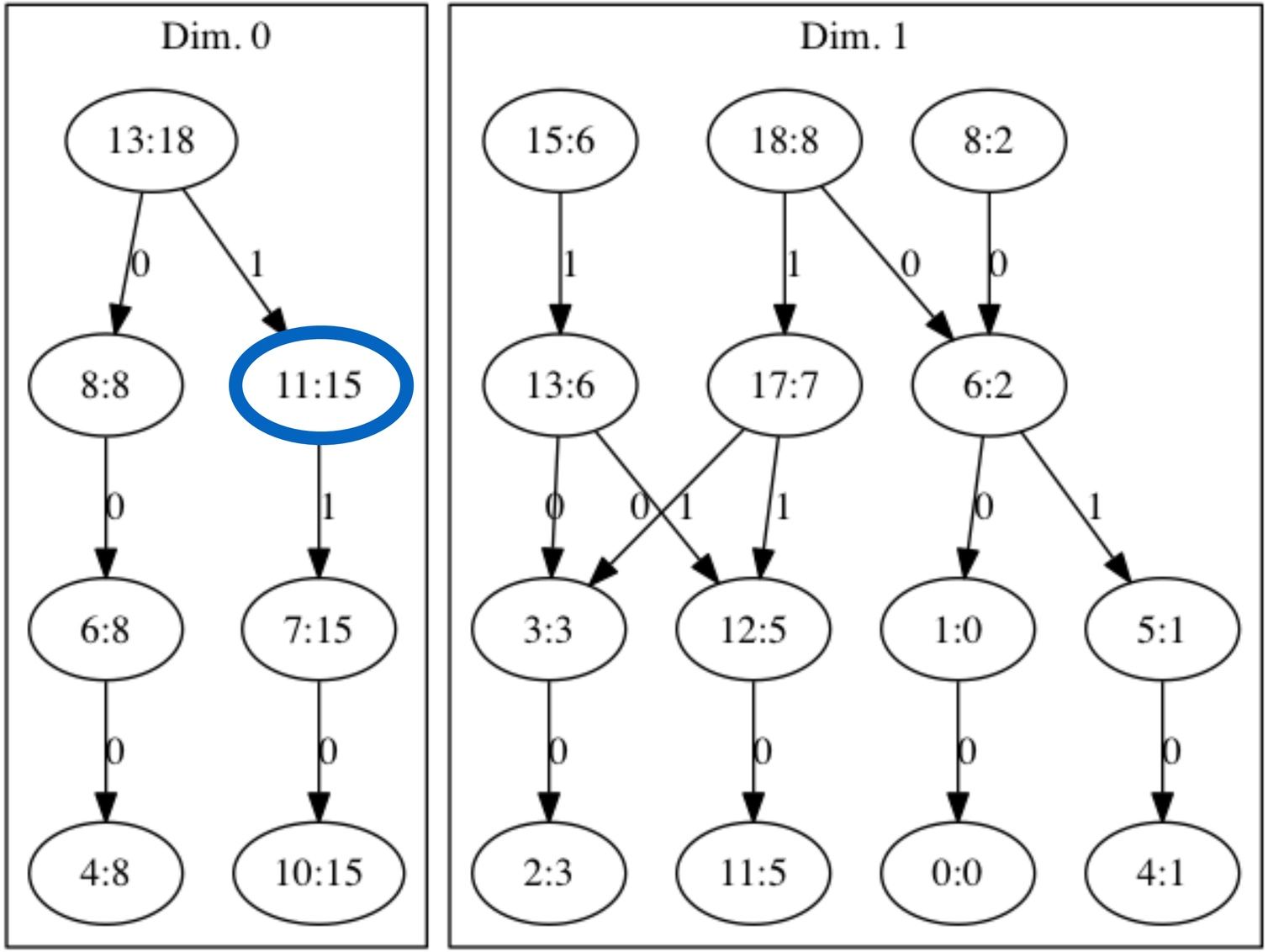
(000, 000)
(000, 010)
(110, 100)
(110, 110)



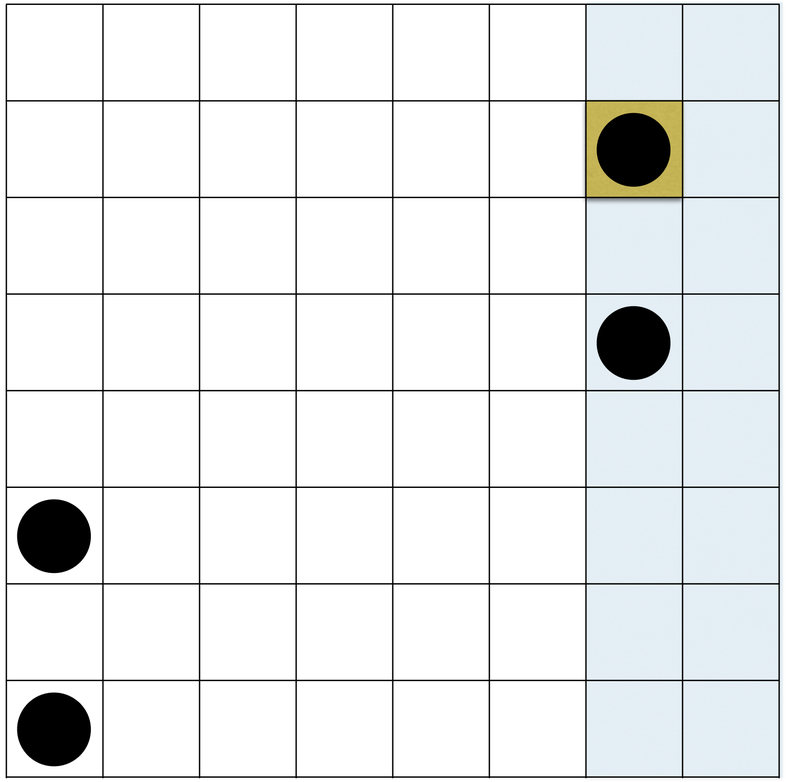
Example 1



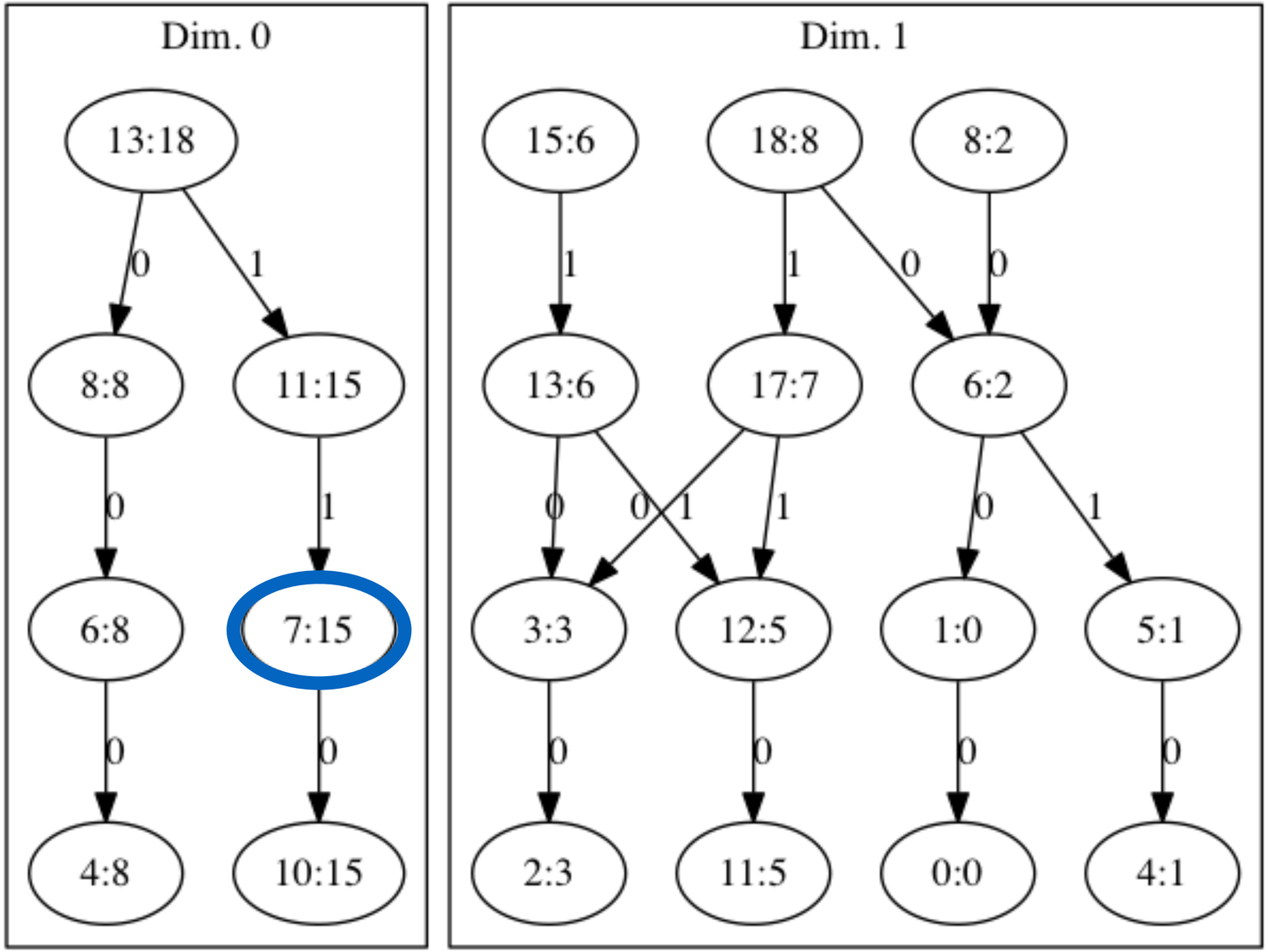
(000, 000)
(000, 010)
(110, 100)
(110, 110)



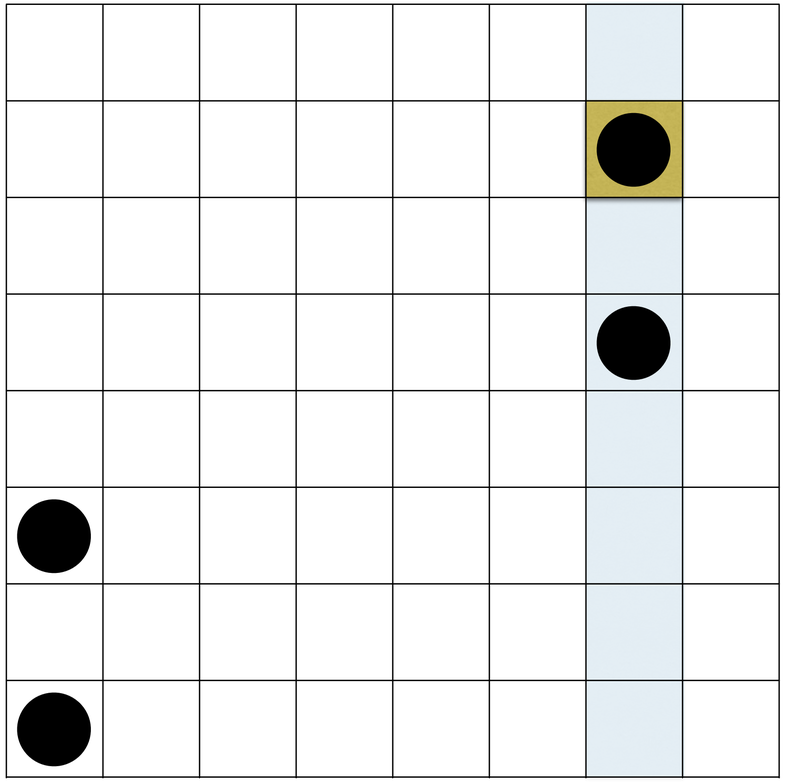
Example 1



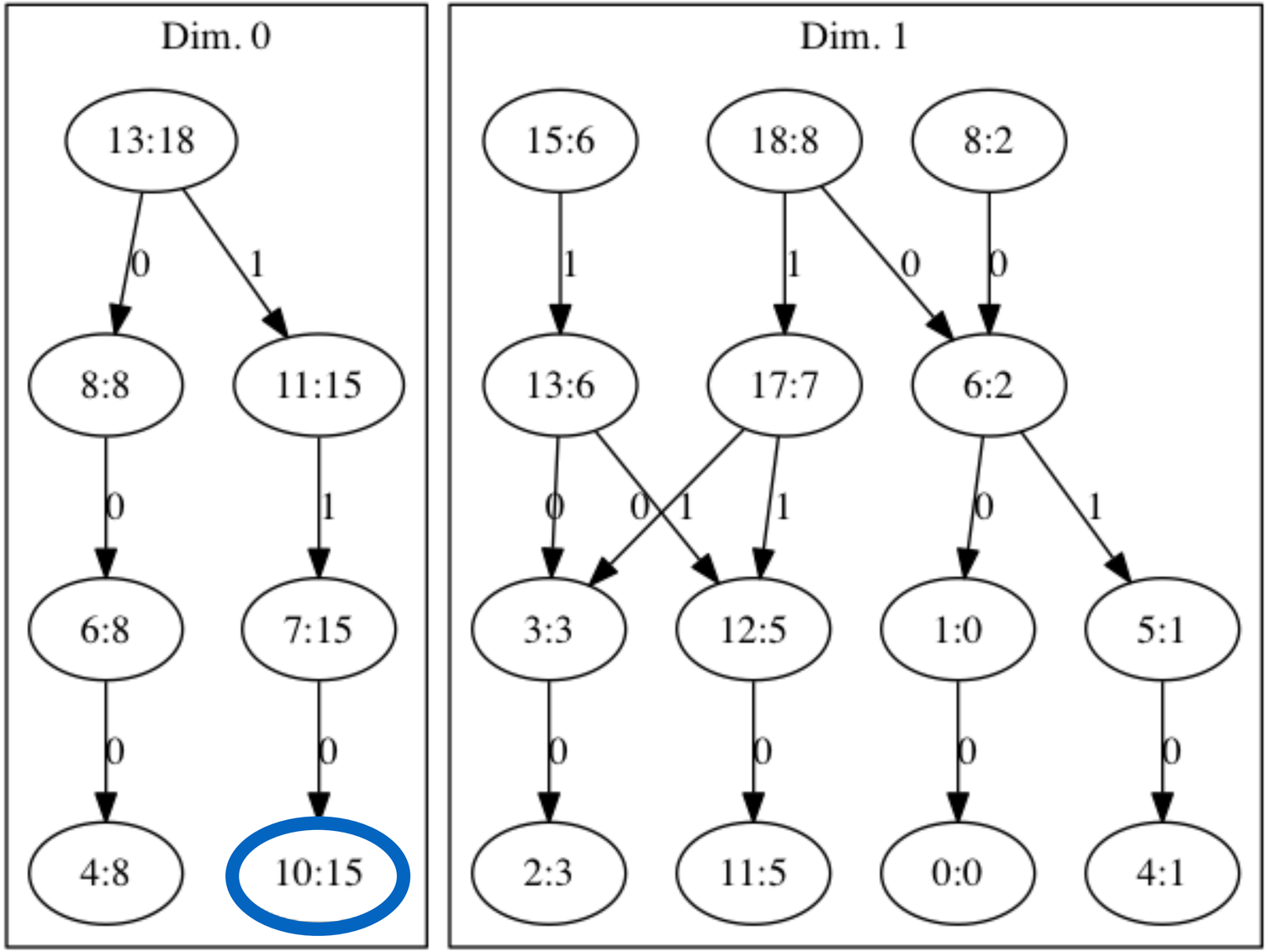
(000, 000)
(000, 010)
(110, 100)
(110, 110)



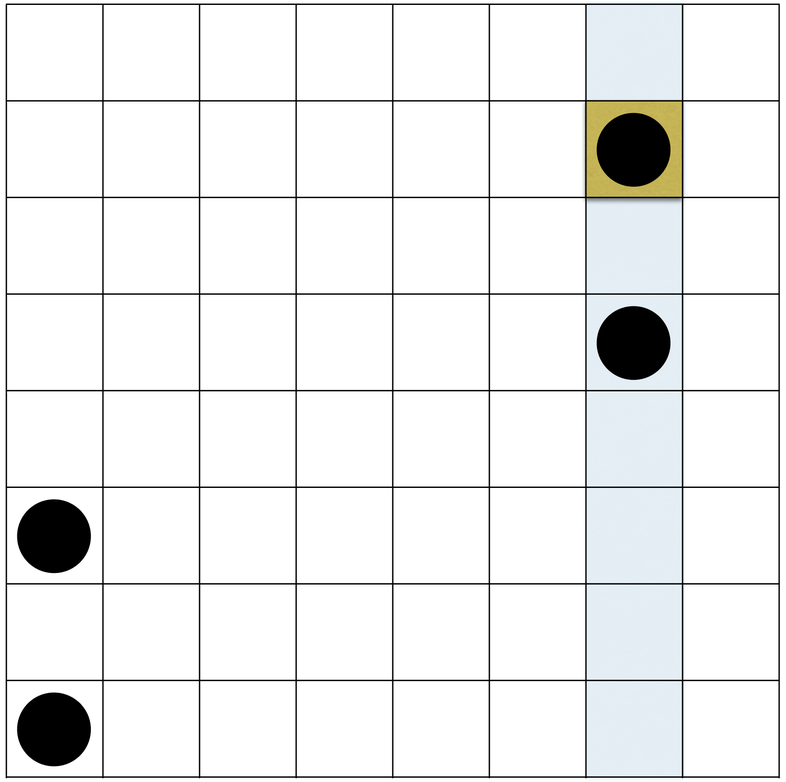
Example 1



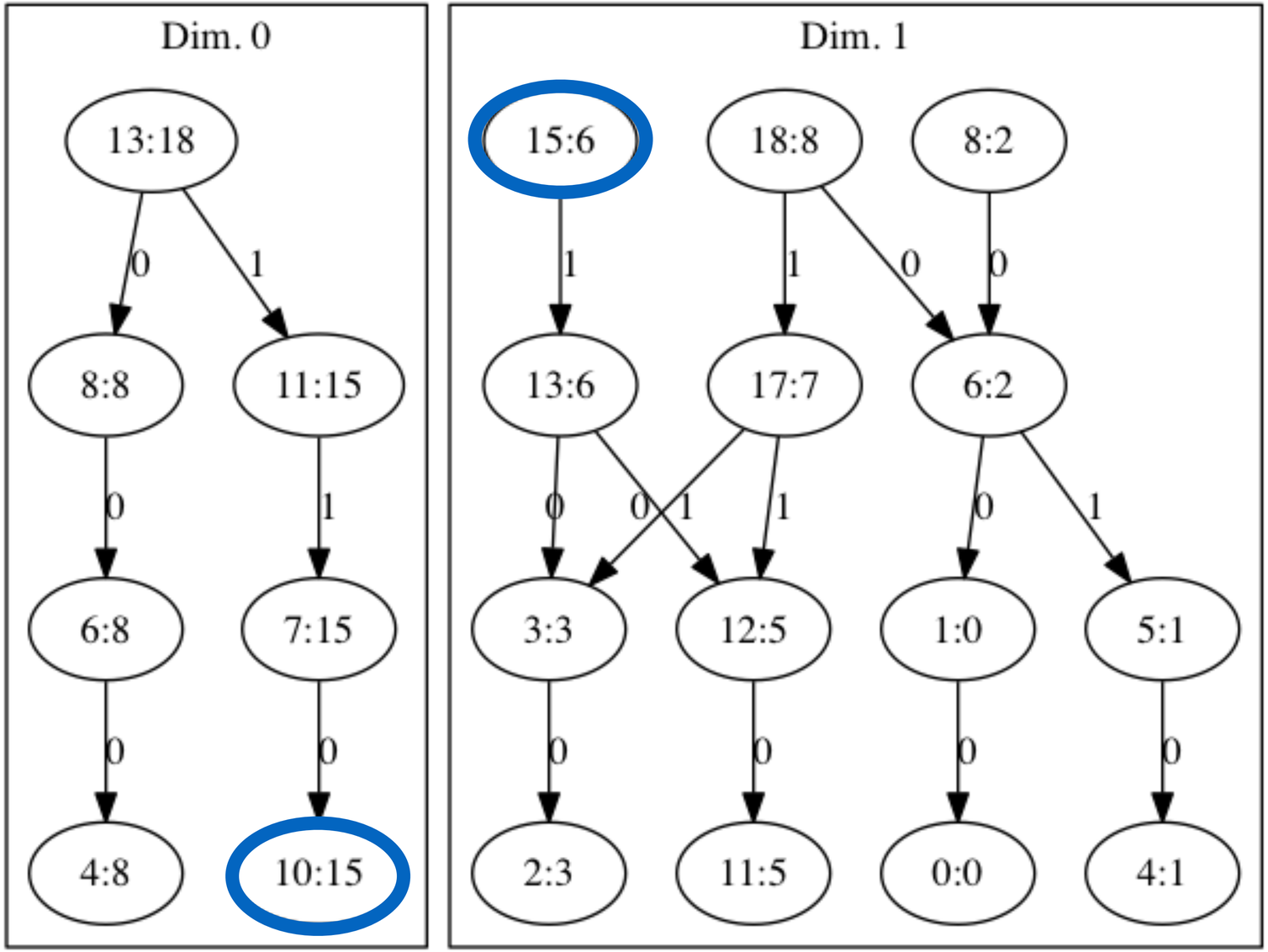
(000, 000)
(000, 010)
(110, 100)
(110, 110)



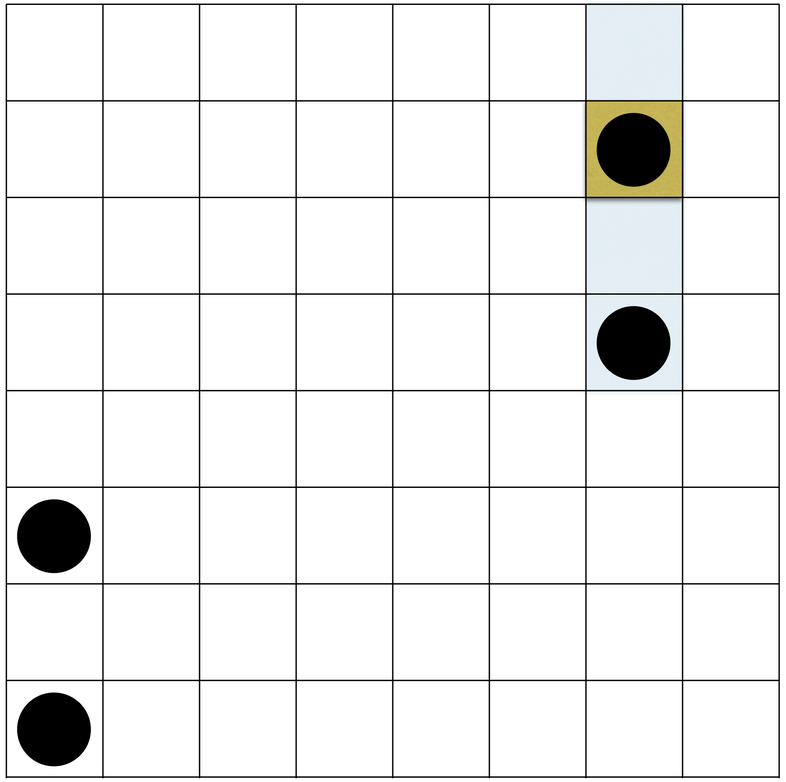
Example 1



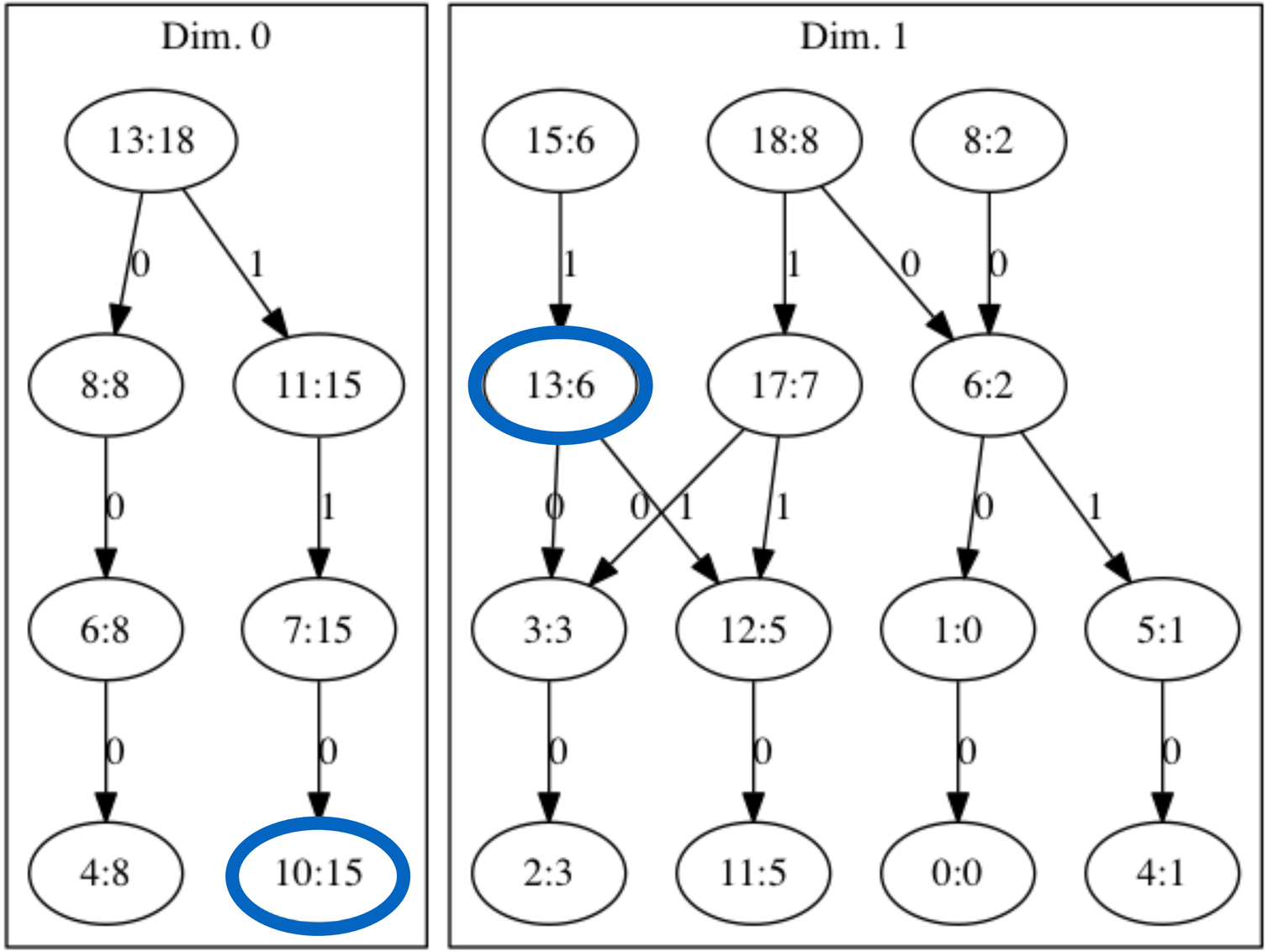
(000, 000)
(000, 010)
(110, 100)
(110, 110)



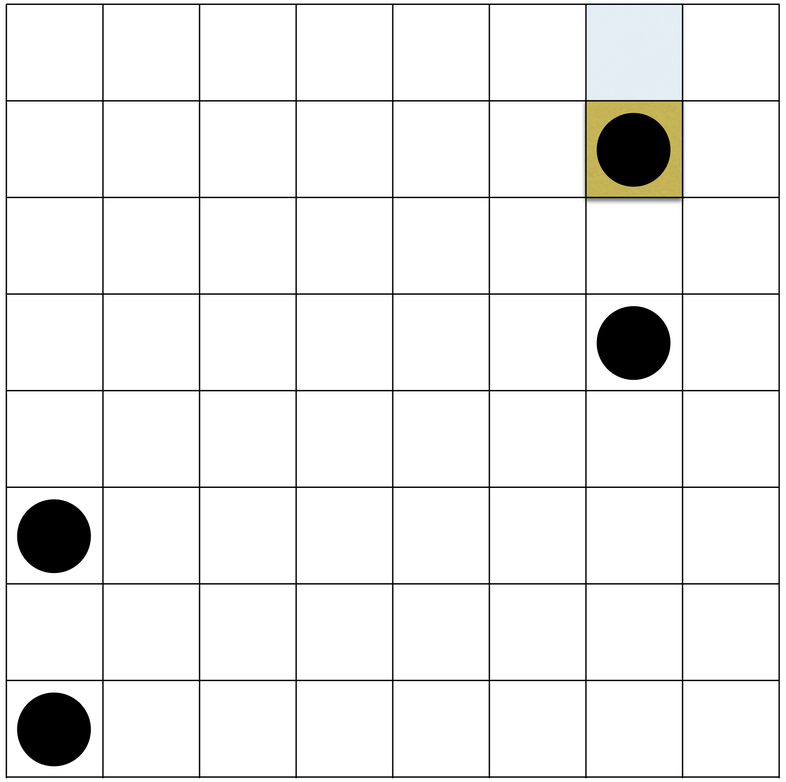
Example 1



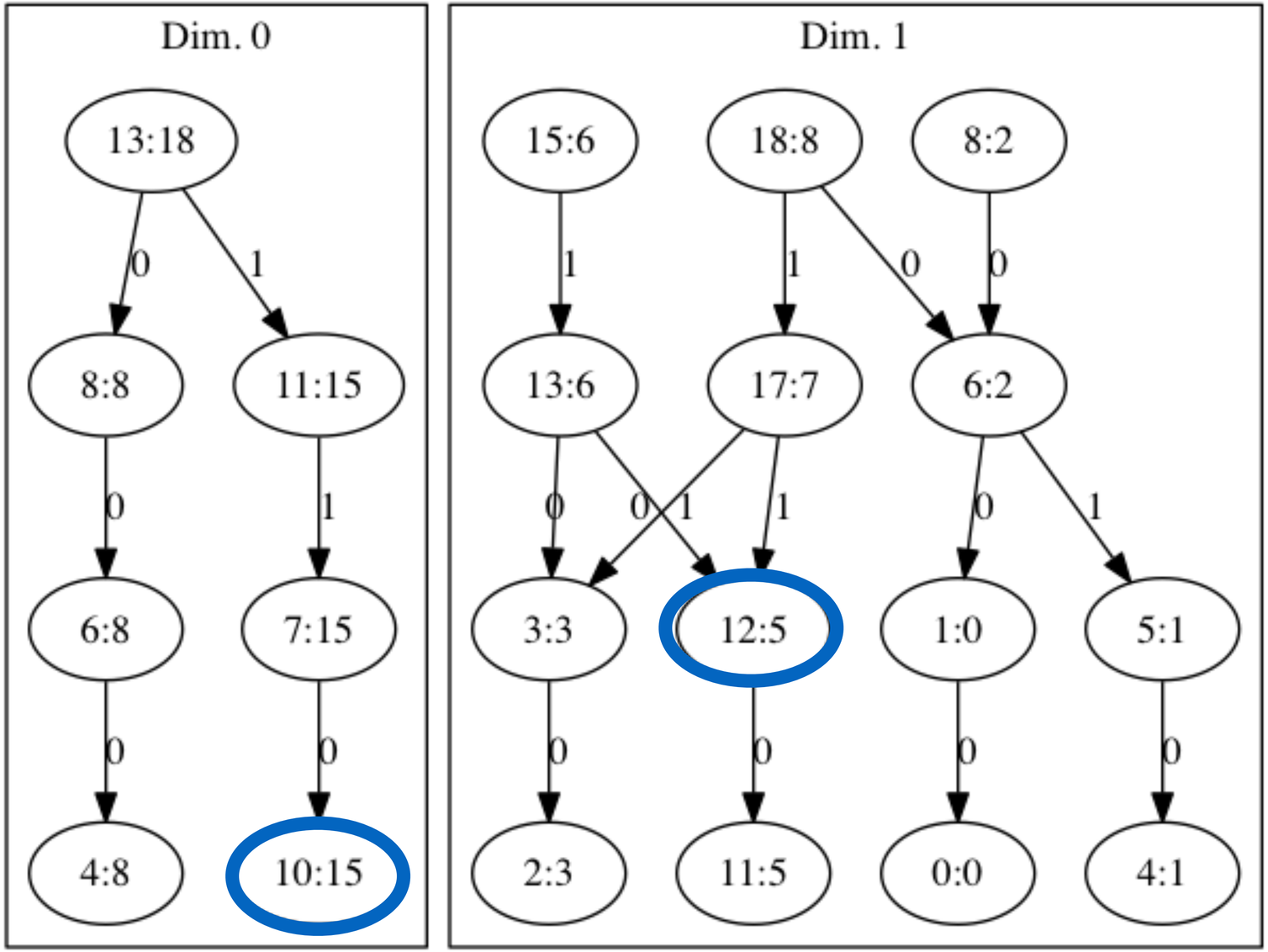
(000, 000)
(000, 010)
(110, 100)
(110, 110)



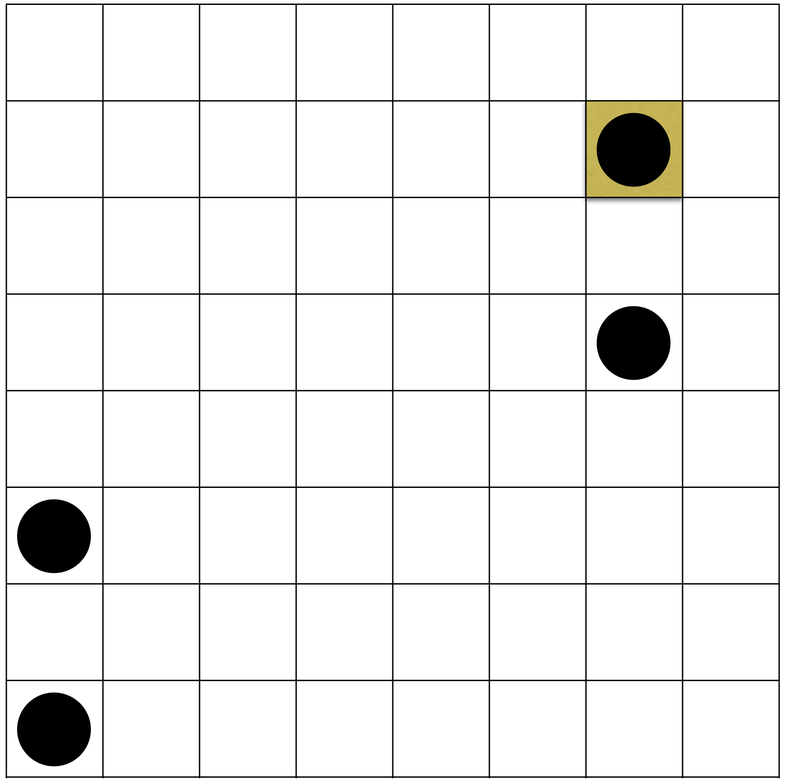
Example 1



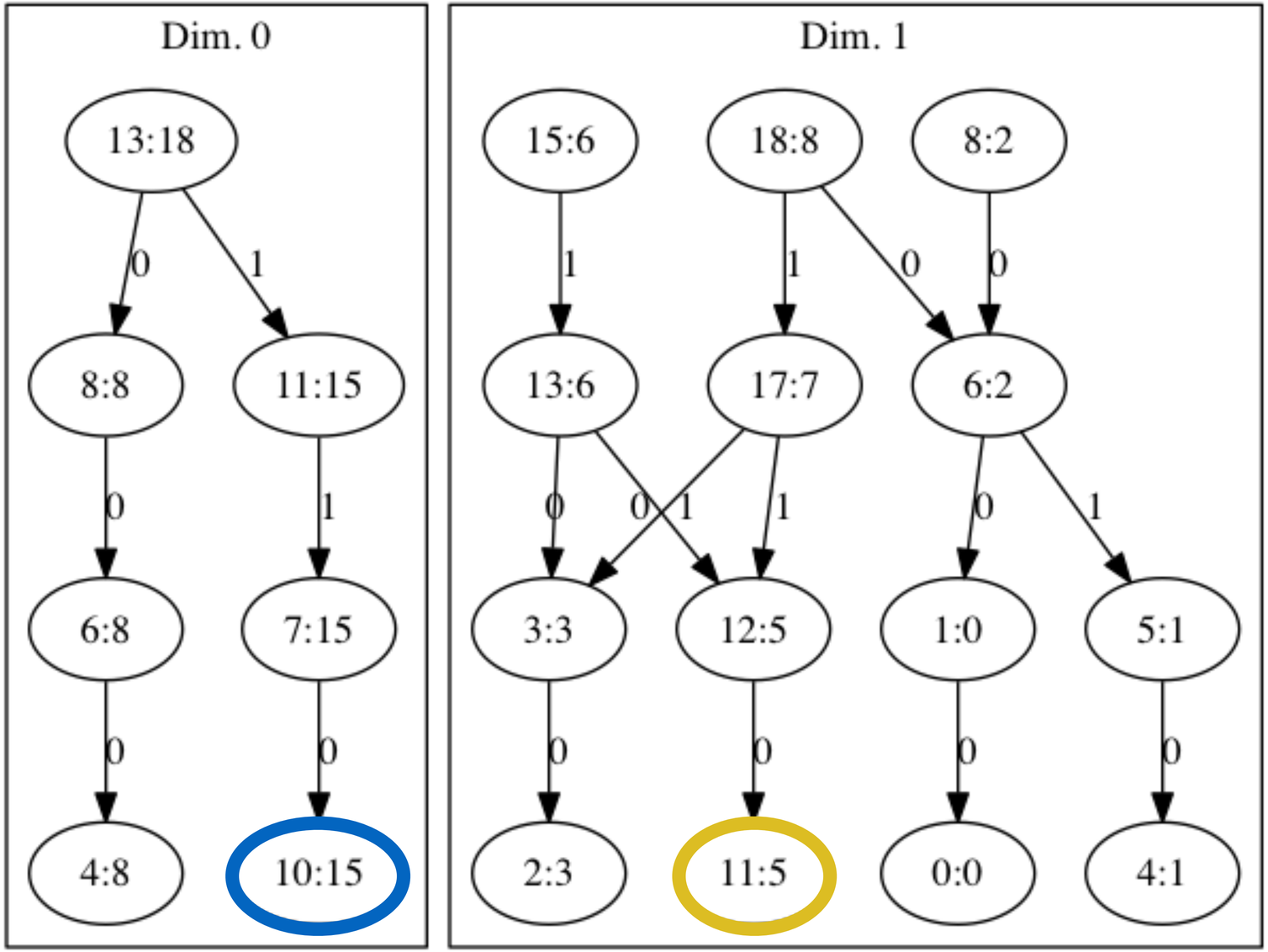
(000, 000)
(000, 010)
(110, 100)
(110, 110)



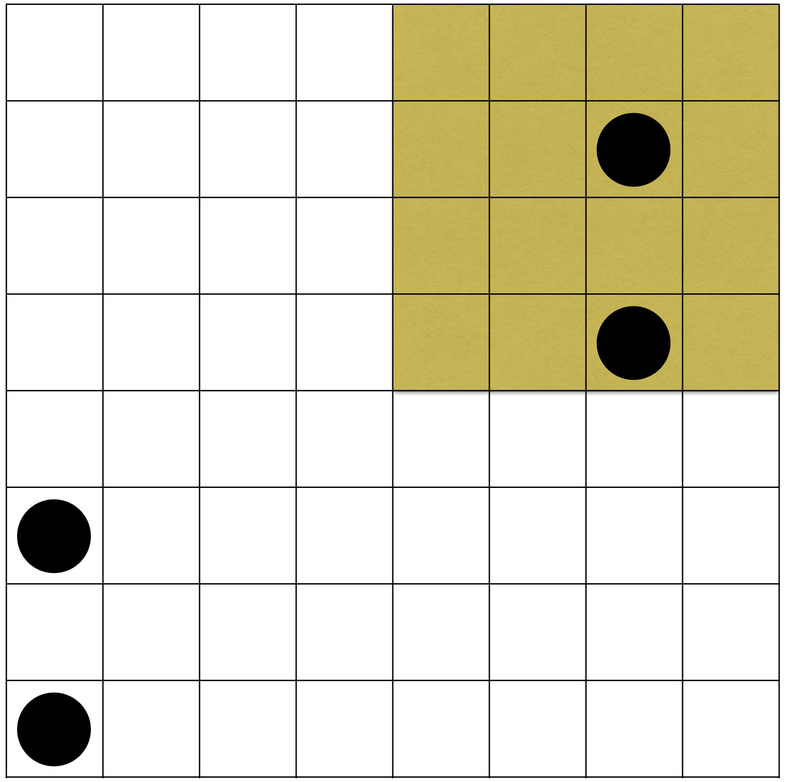
Example 1



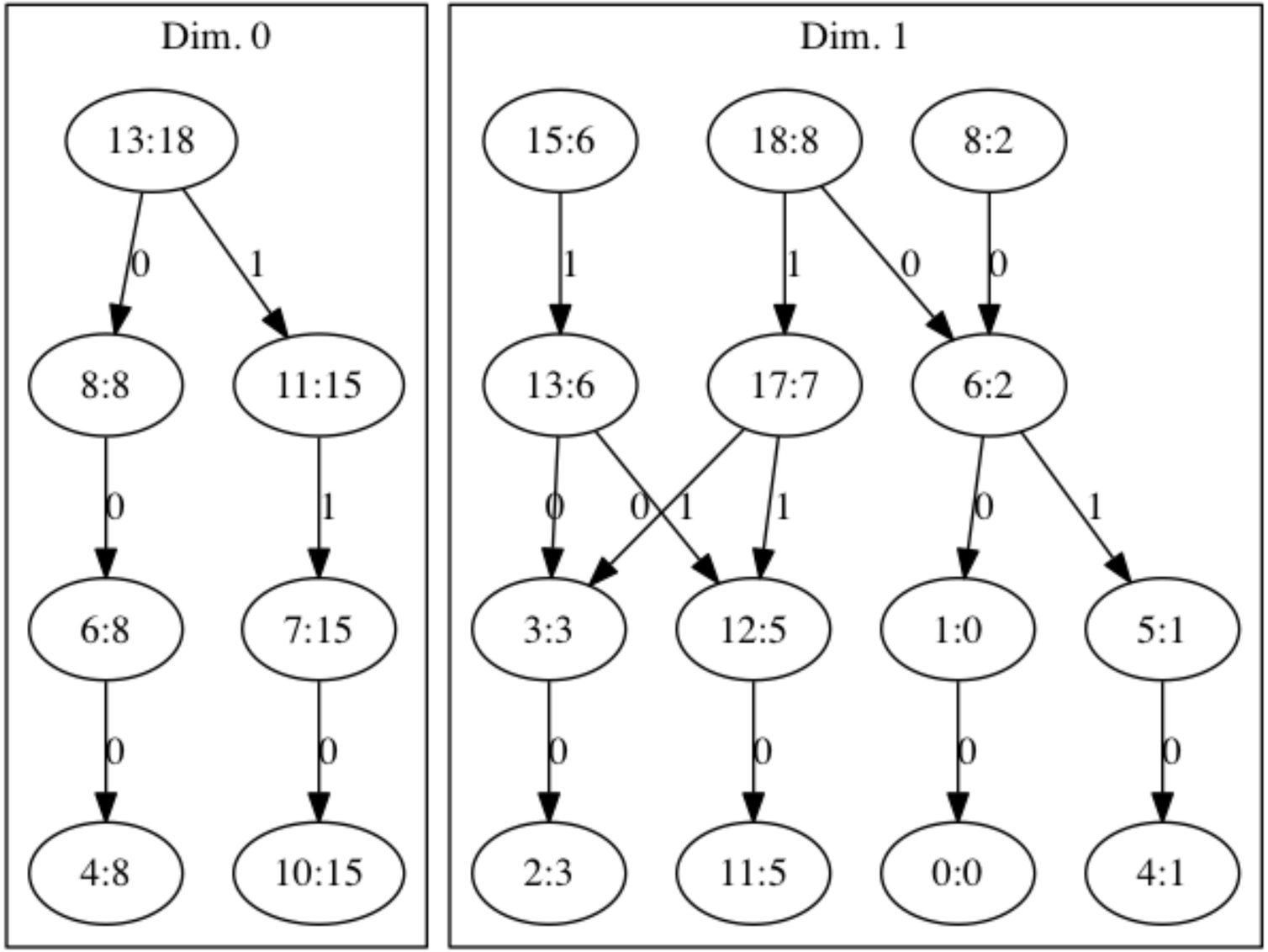
(000, 000)
(000, 010)
(110, 100)
(110, 110)



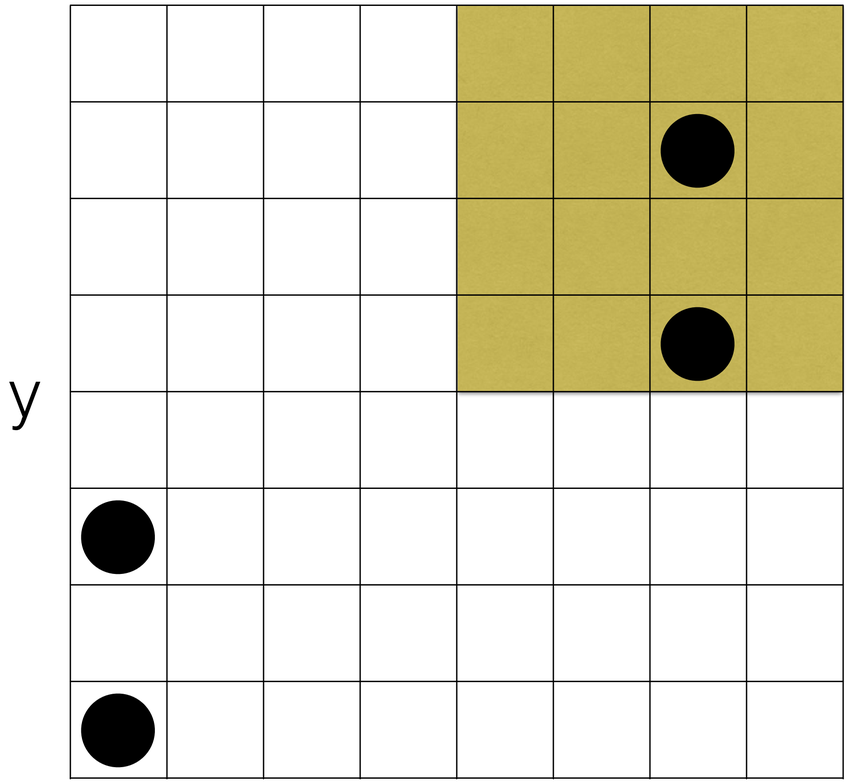
Example 1b



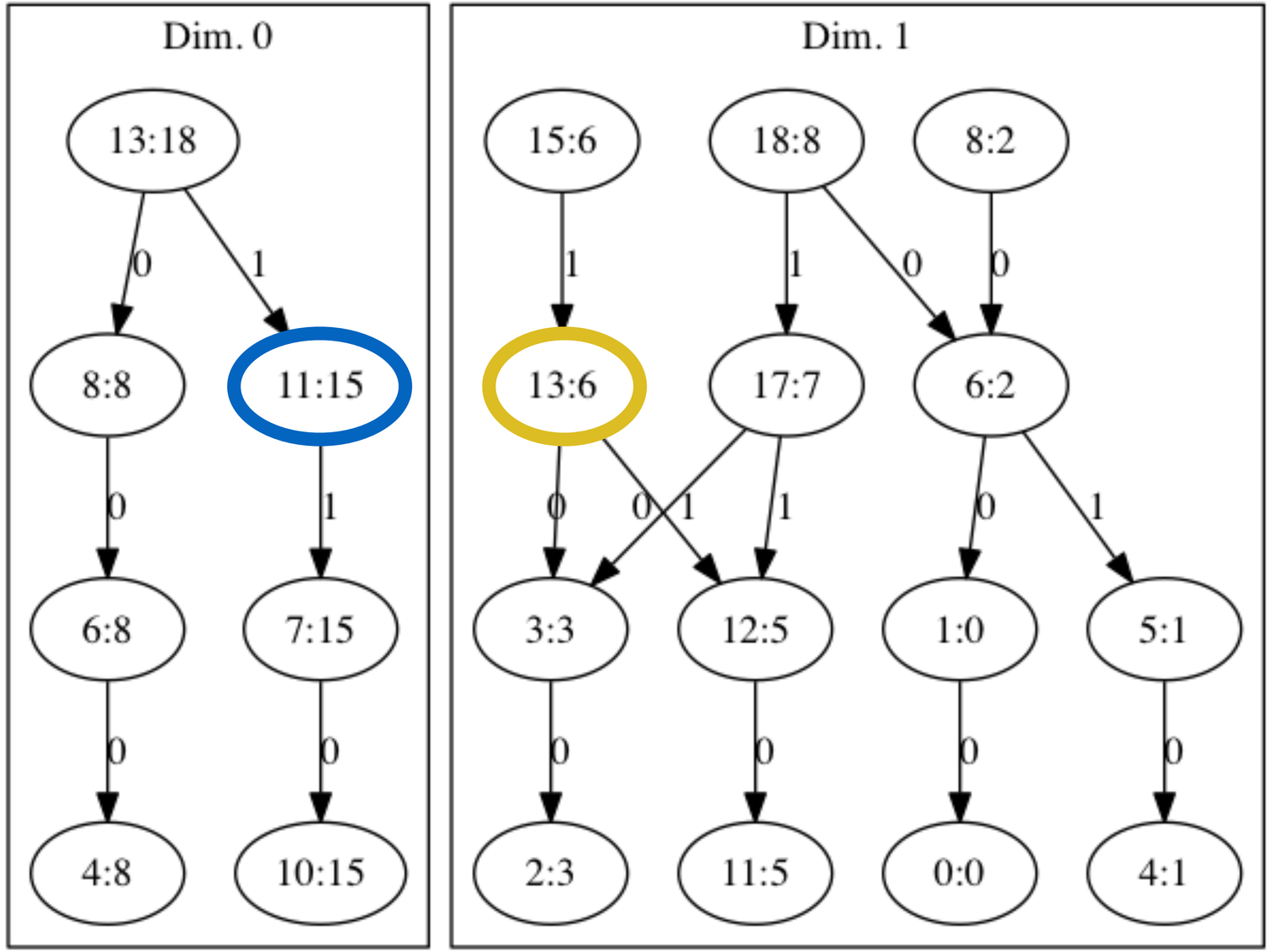
(000, 000)
(000, 010)
(110, 100)
(110, 110)



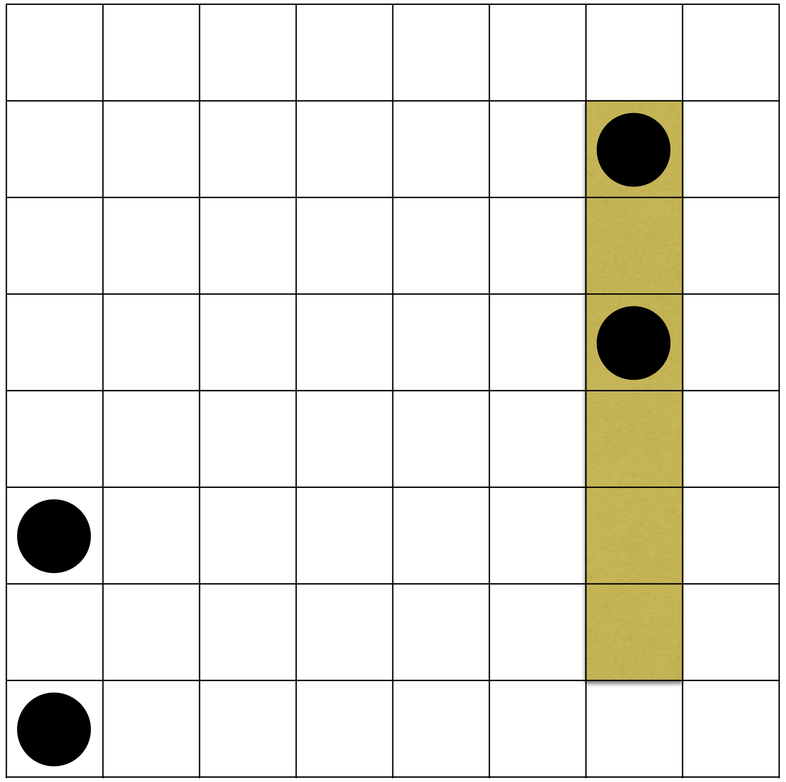
Example 1b



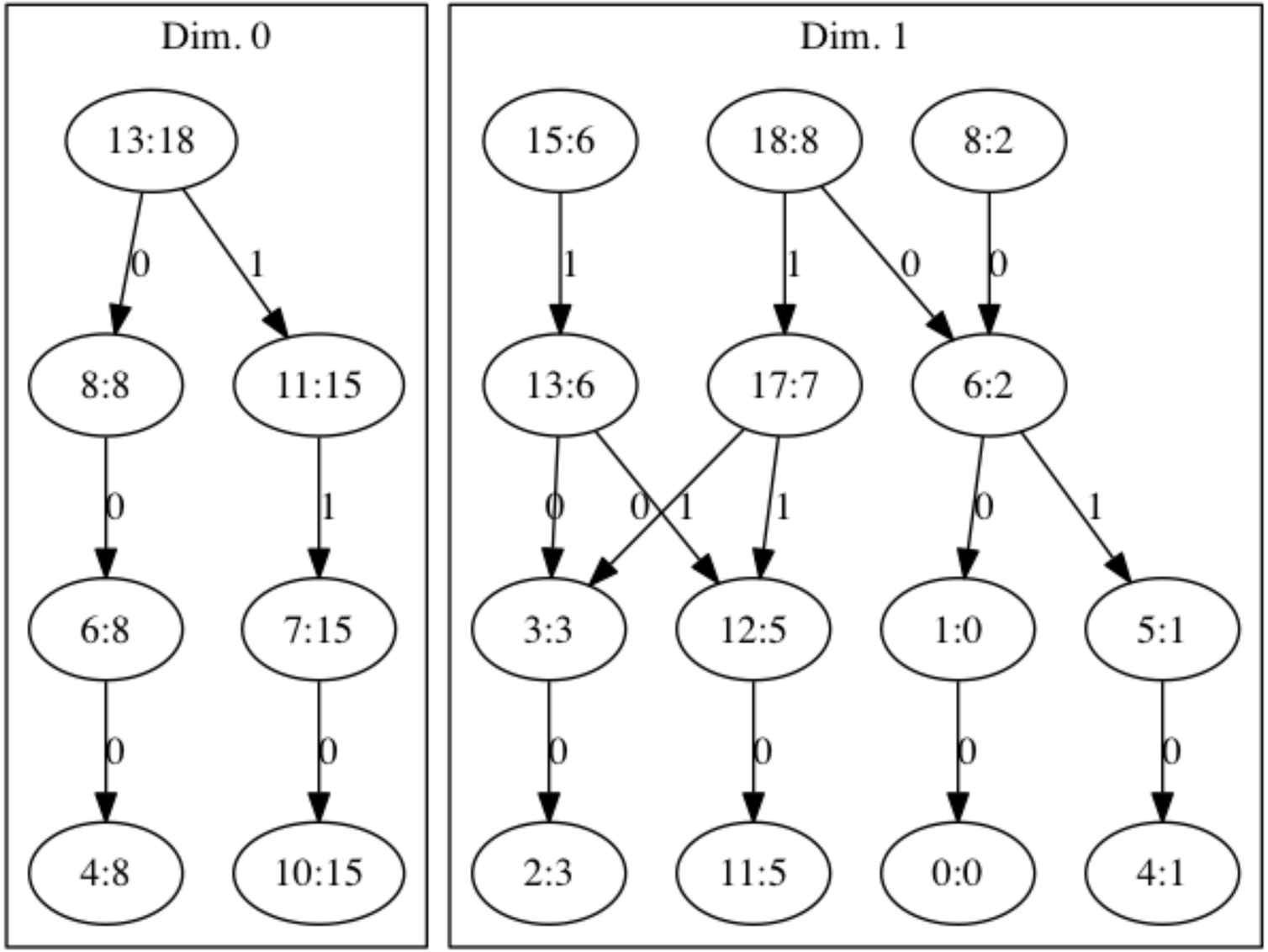
(000, 000)
(000, 010)
(110, 100)
(110, 110)



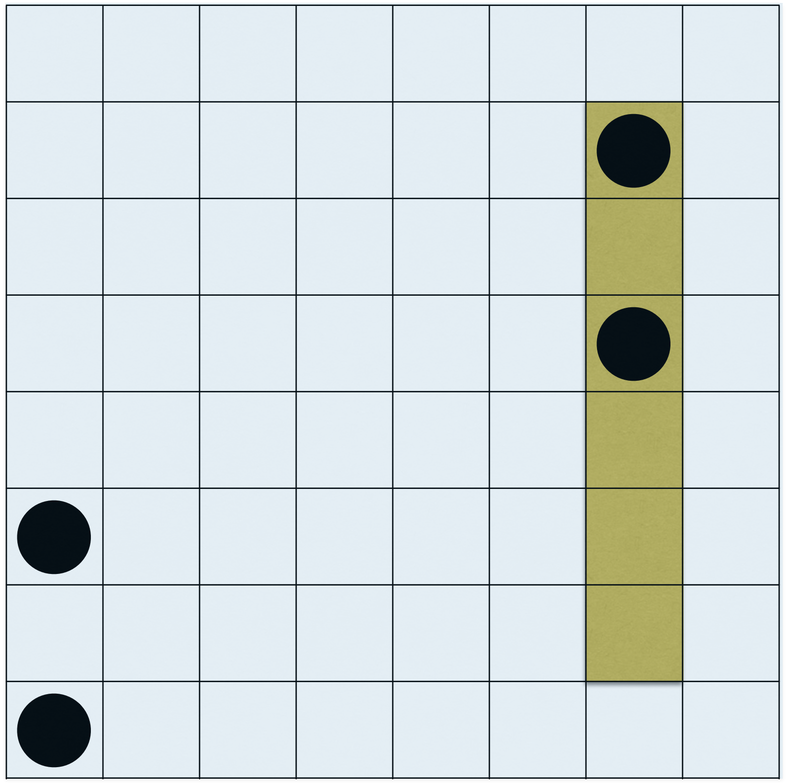
Example 2



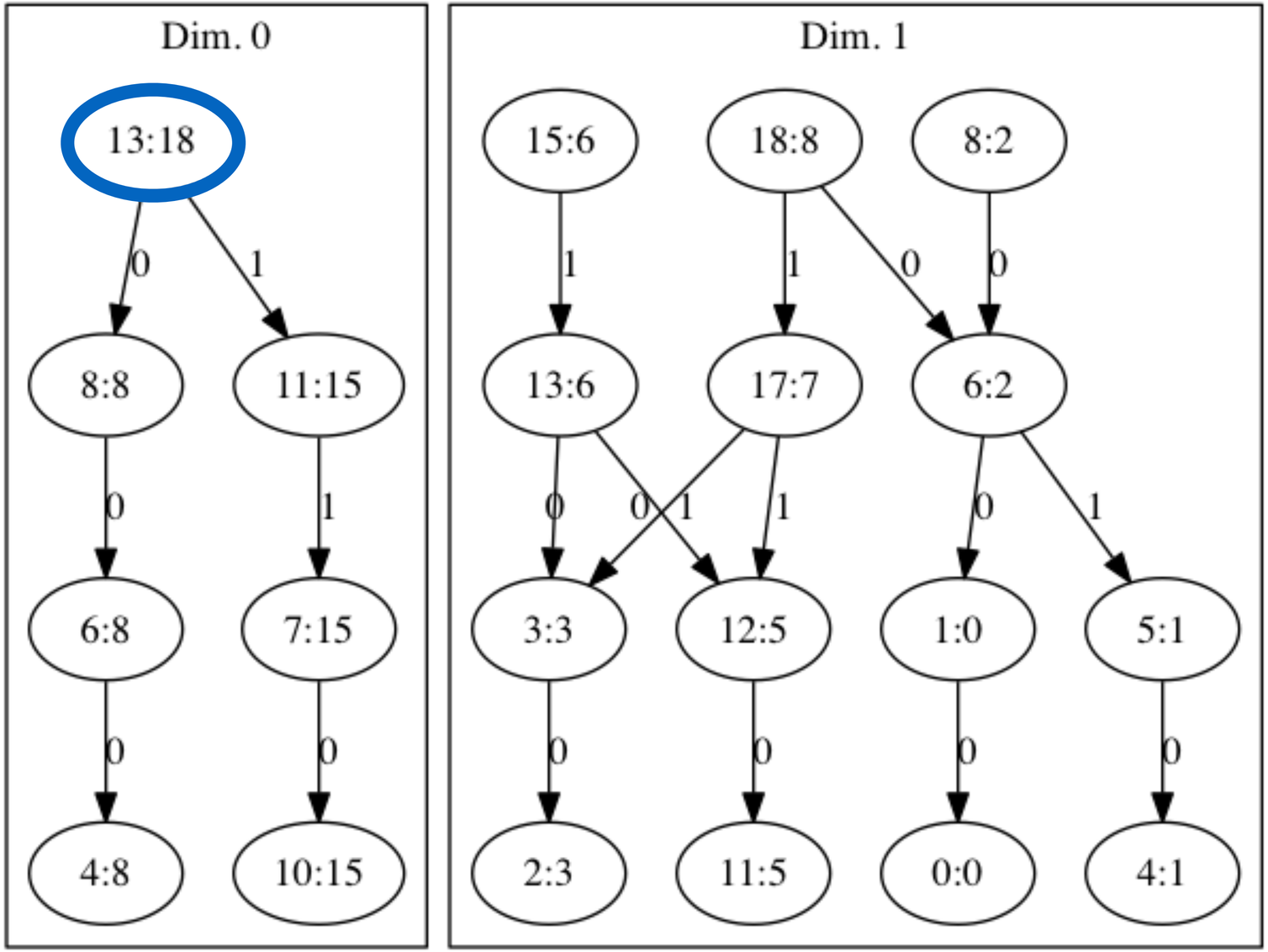
(000, 000)
(000, 010)
(110, 100)
(110, 110)



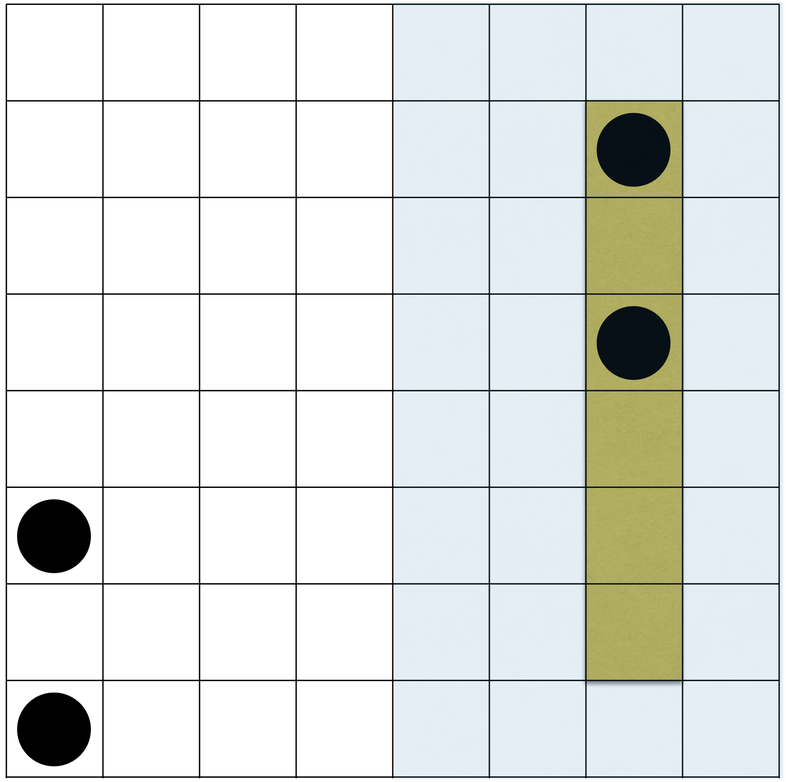
Example 2



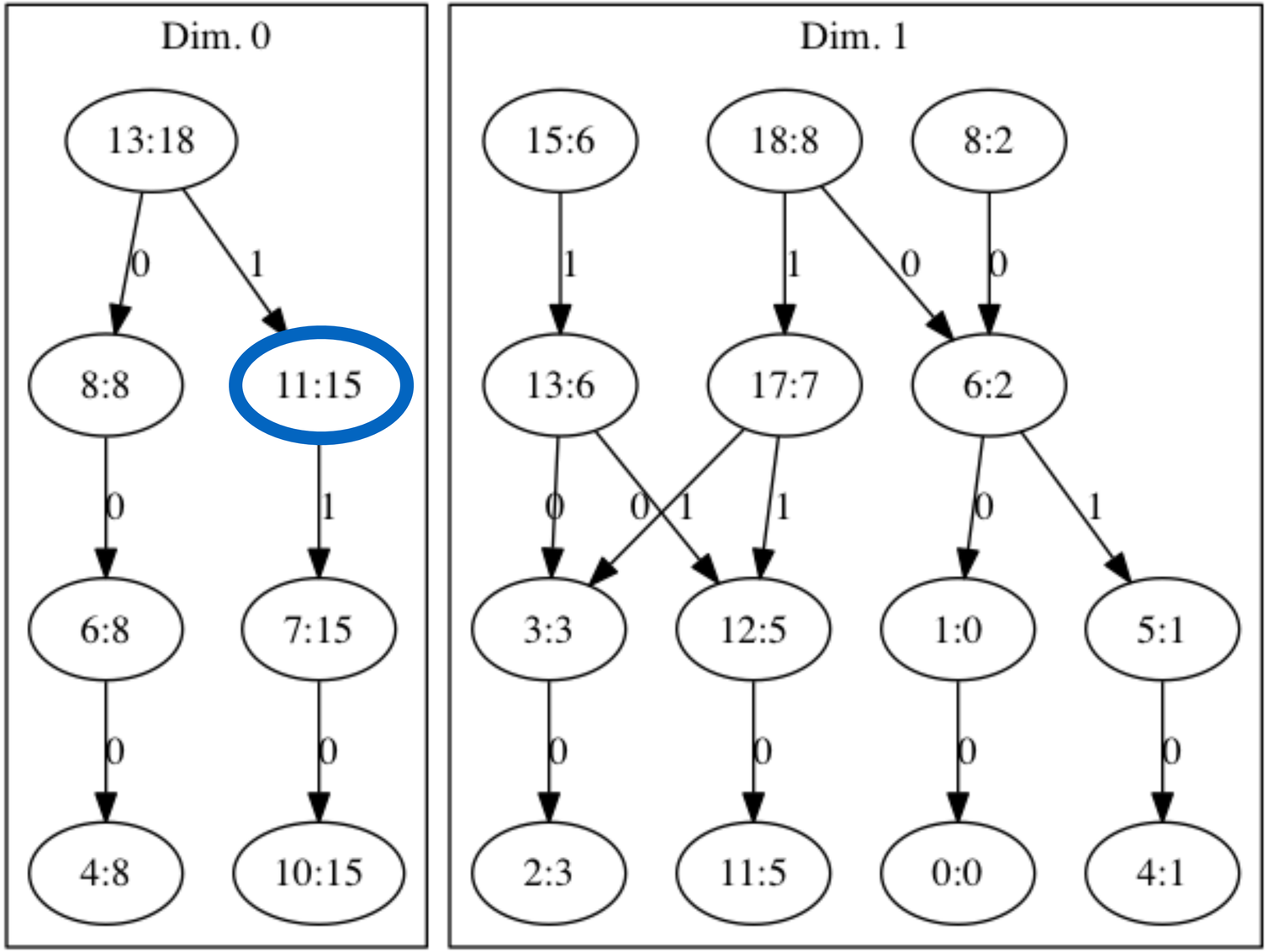
(000, 000)
(000, 010)
(110, 100)
(110, 110)



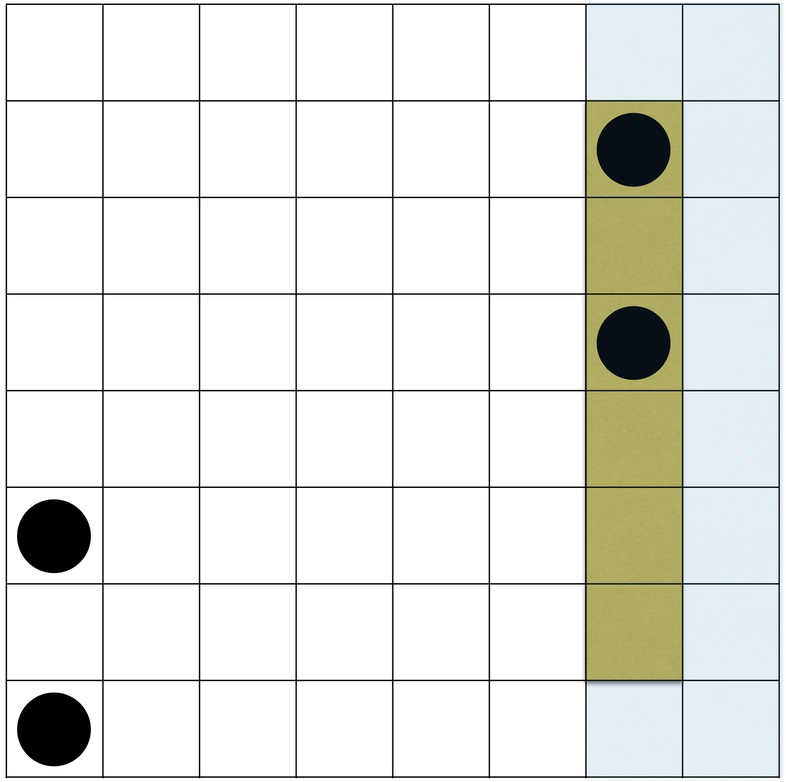
Example 2



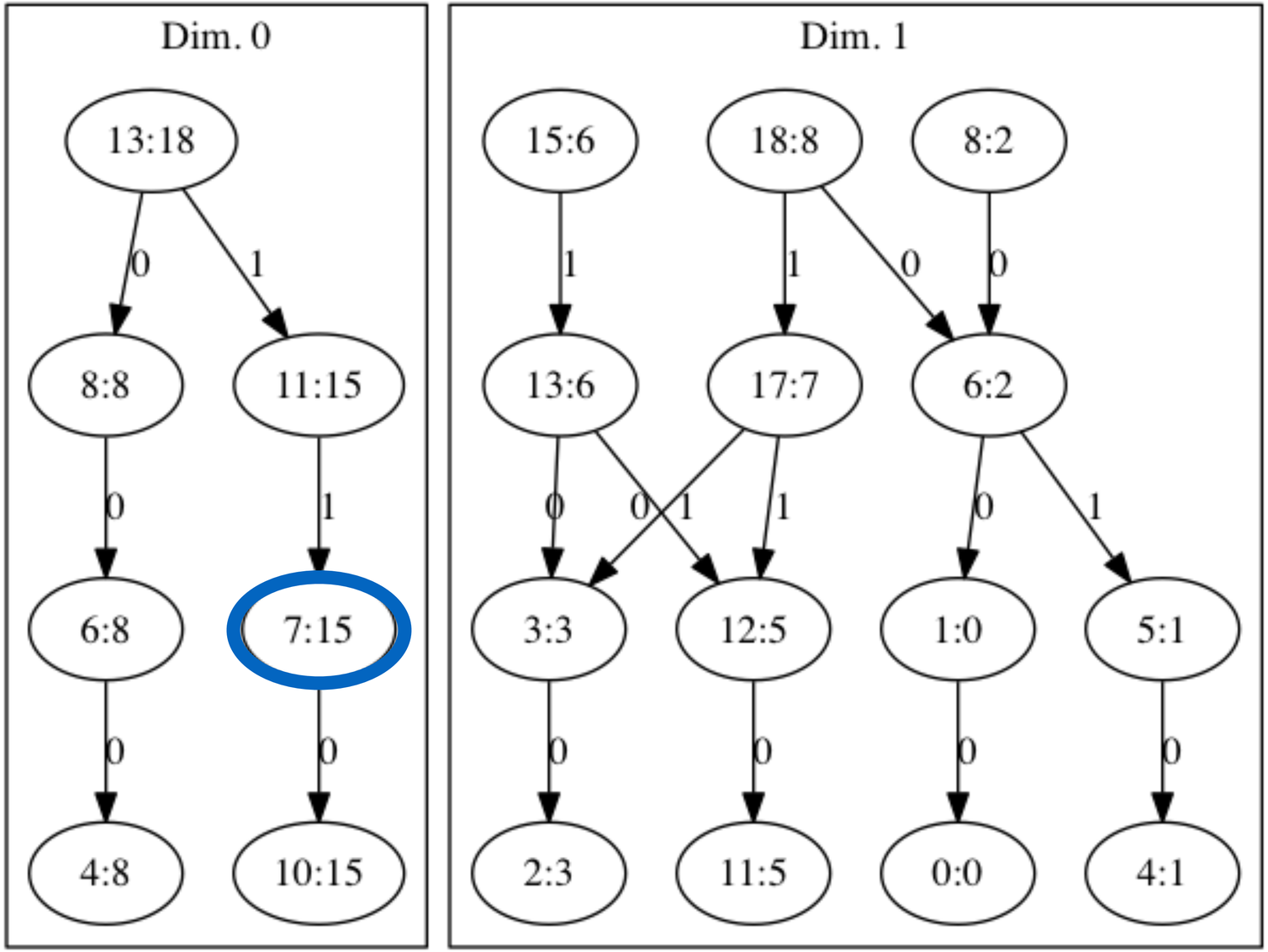
(000, 000)
(000, 010)
(110, 100)
(110, 110)



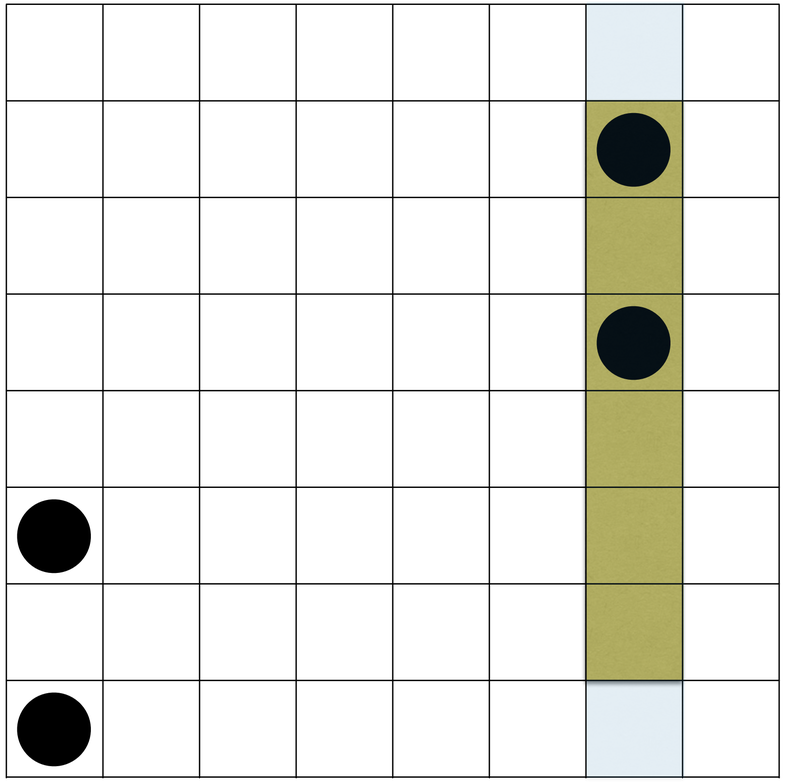
Example 2



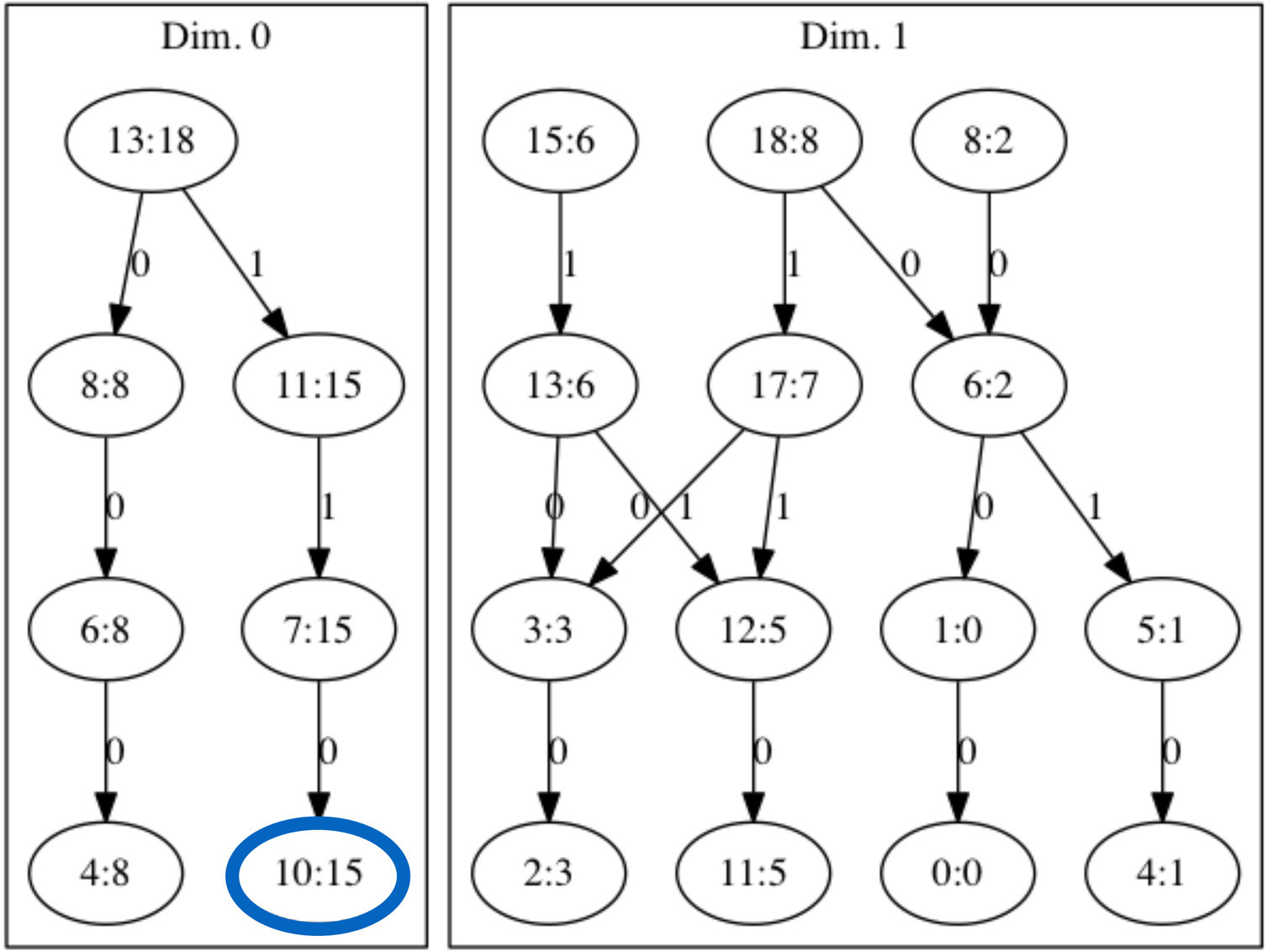
(000, 000)
(000, 010)
(110, 100)
(110, 110)



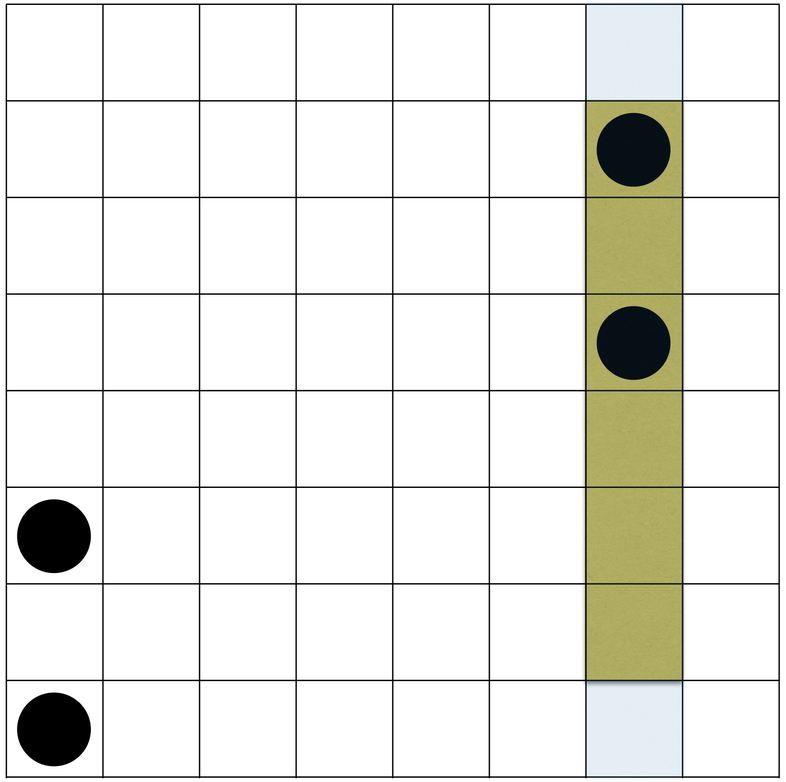
Example 2



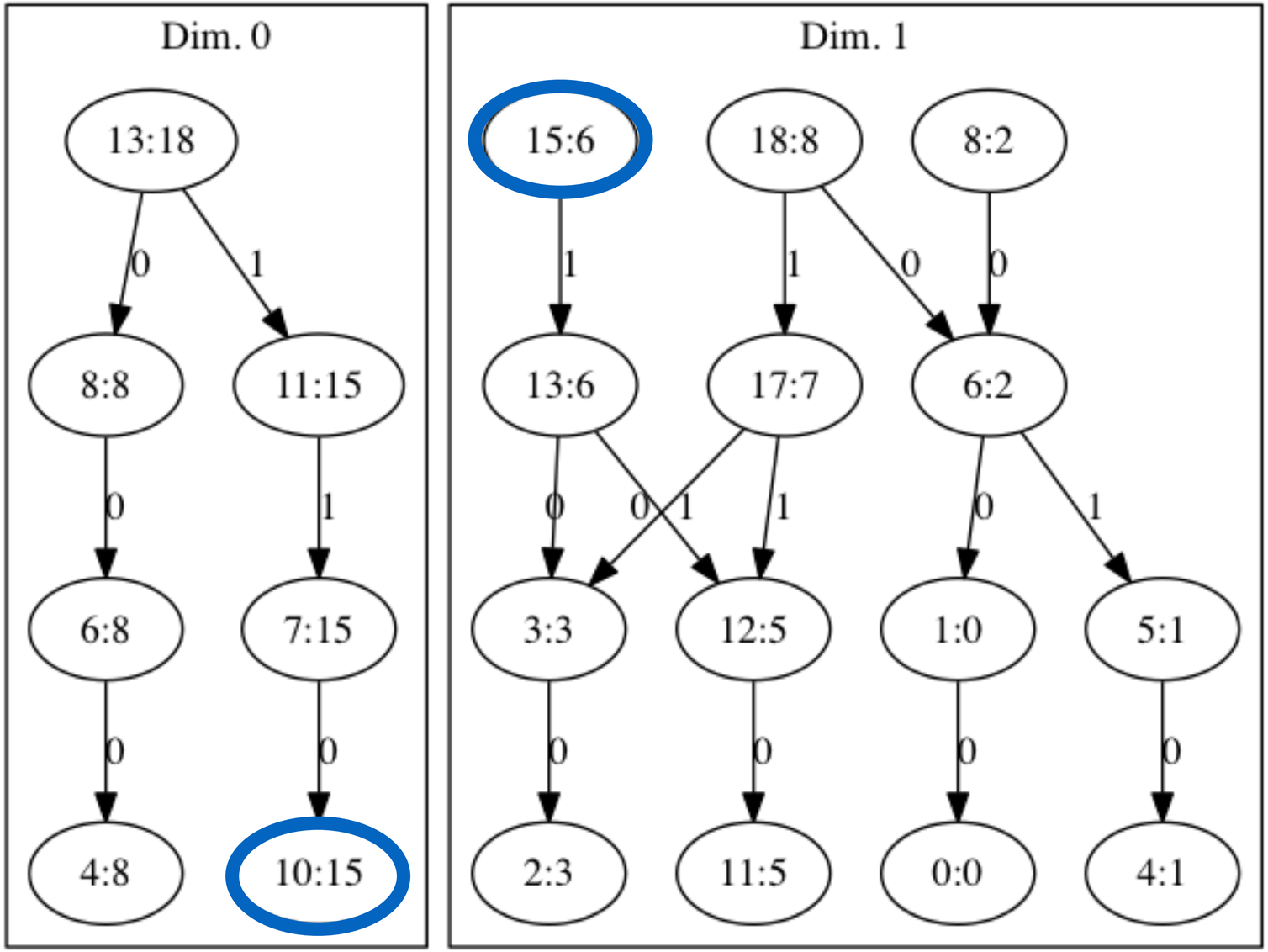
(000, 000)
(000, 010)
(110, 100)
(110, 110)



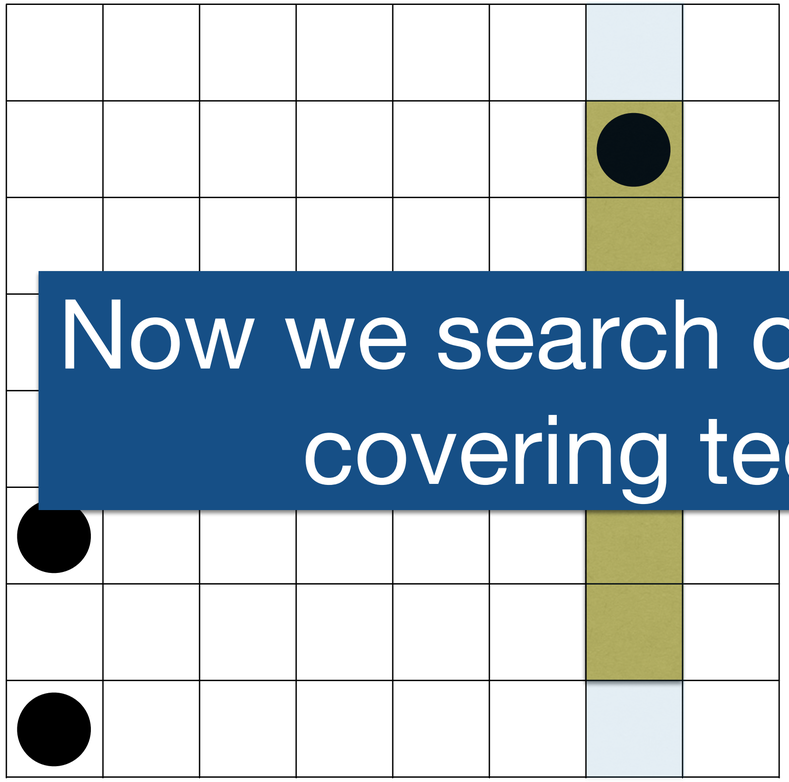
Example 2

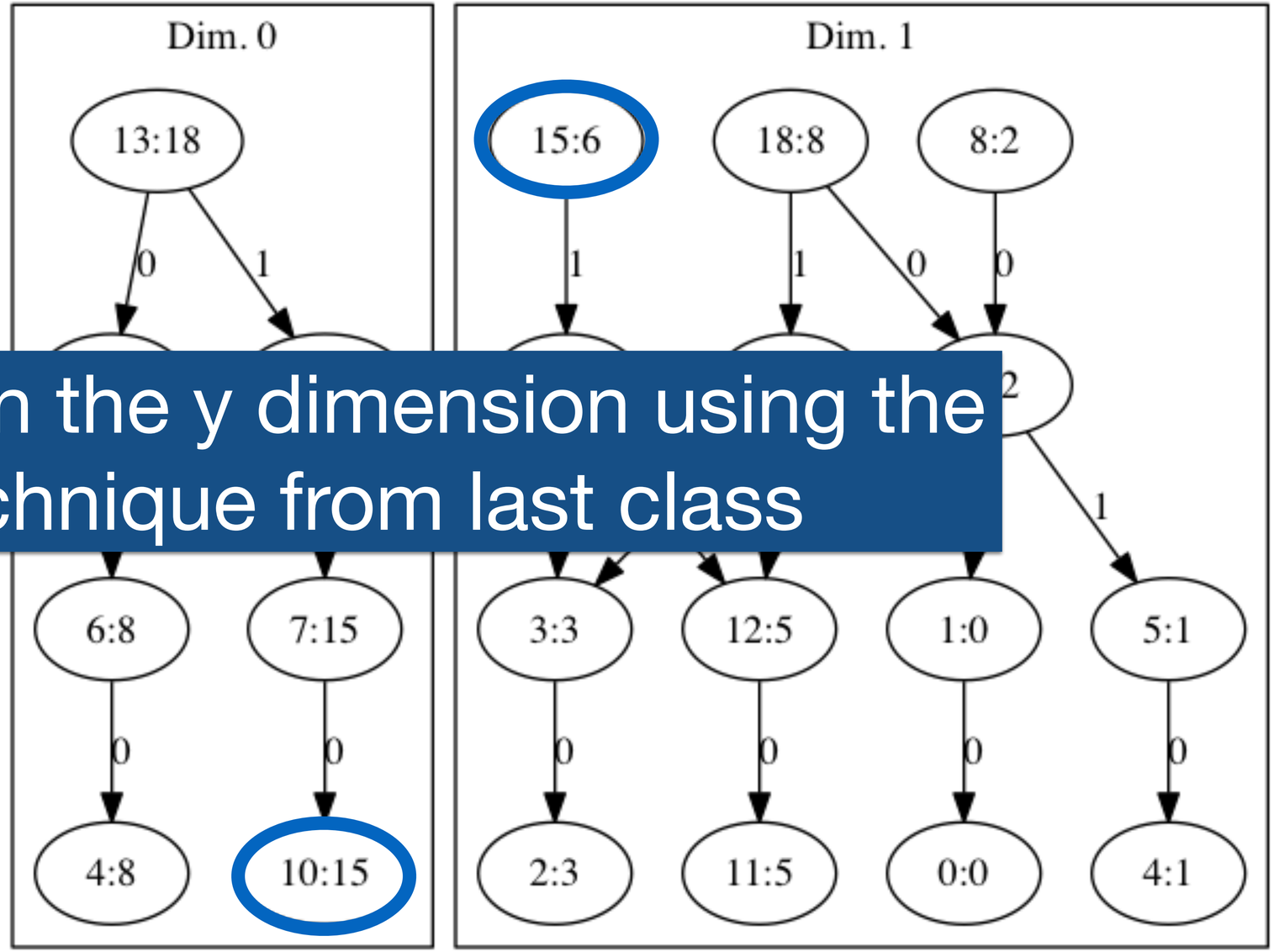


(000, 000)
(000, 010)
(110, 100)
(110, 110)



Example 2

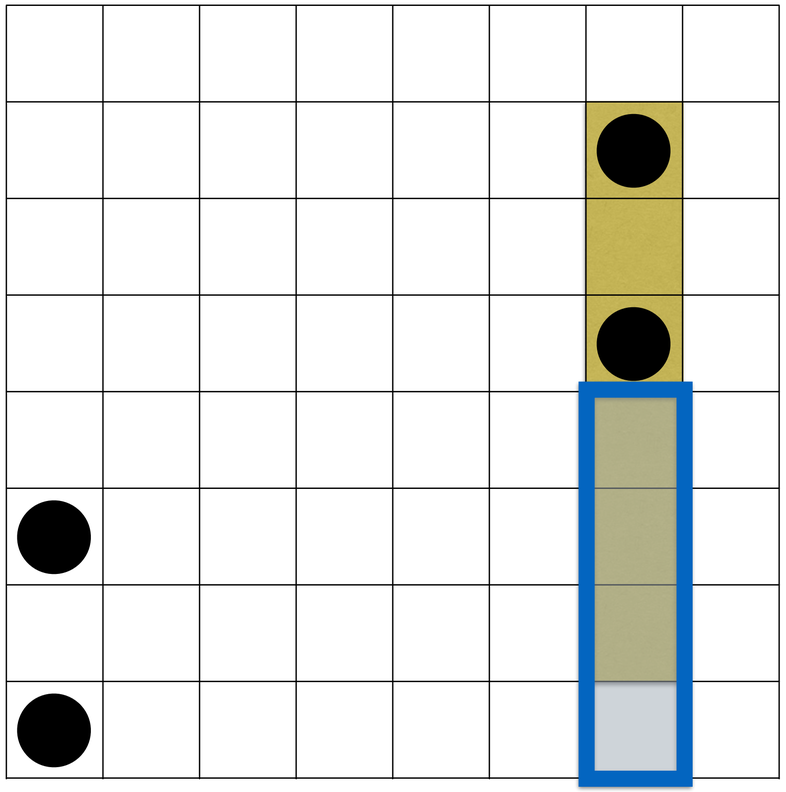




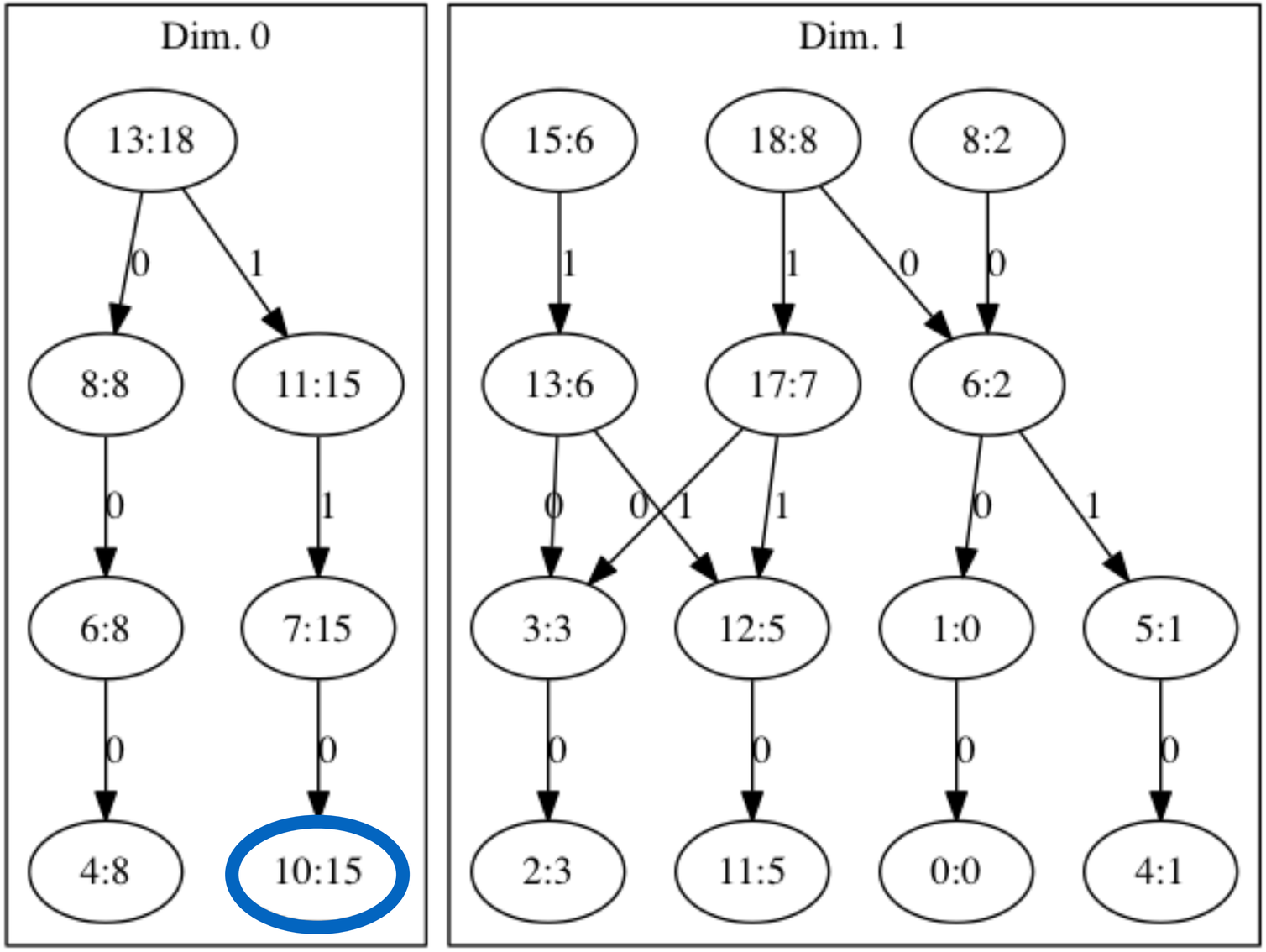
Now we search on the y dimension using the covering technique from last class

(000, 000)
(000, 010)
(110, 100)
(110, 110)

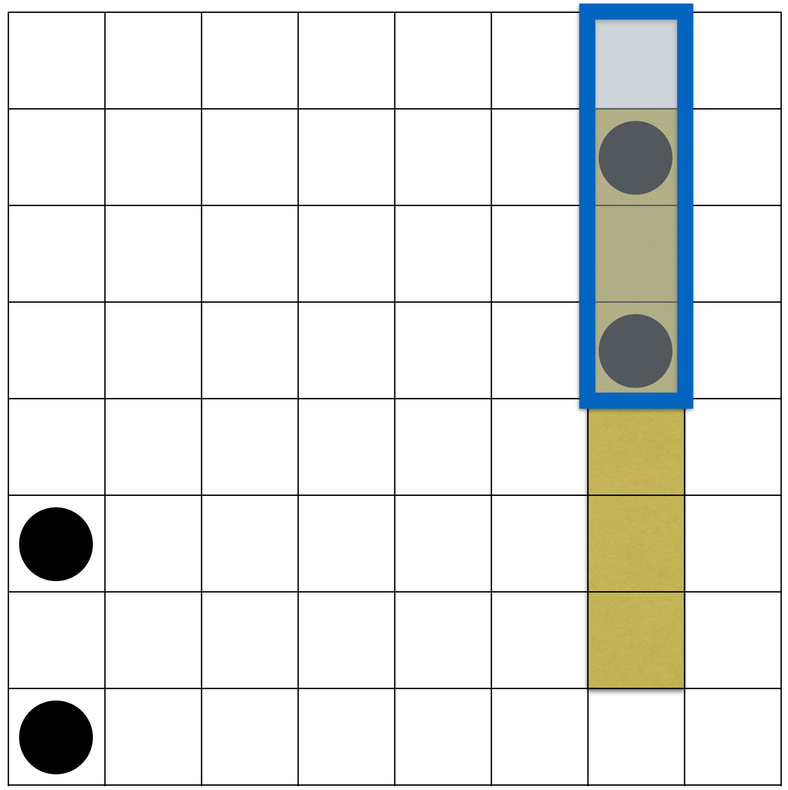
Example 2



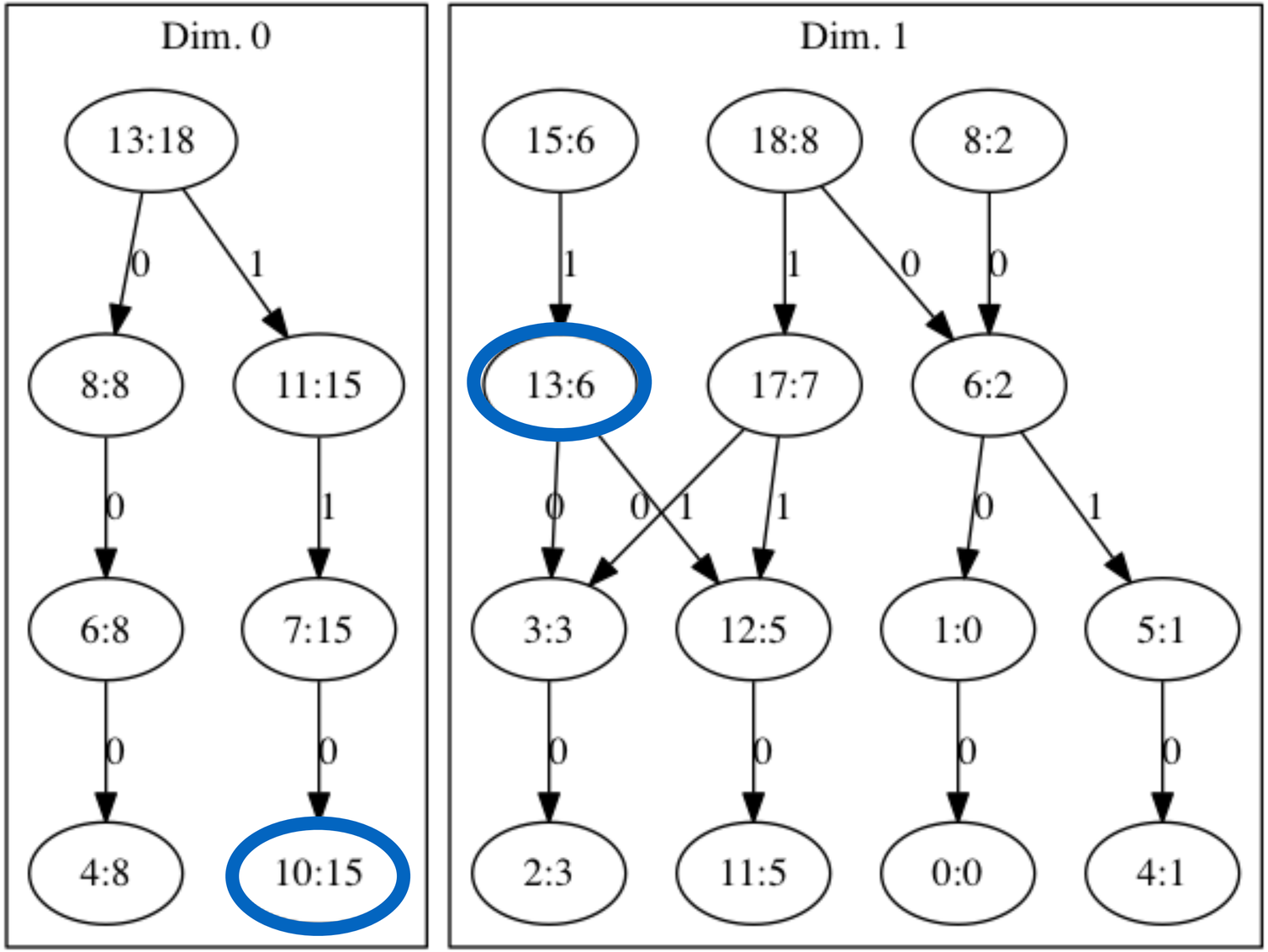
(000, 000)
(000, 010)
(110, 100)
(110, 110)



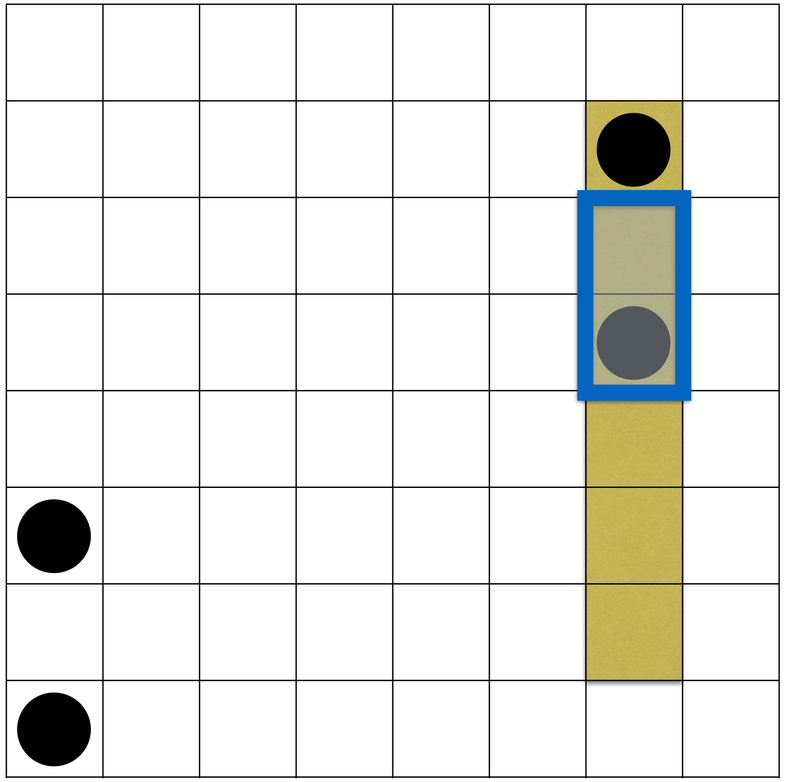
Example 2



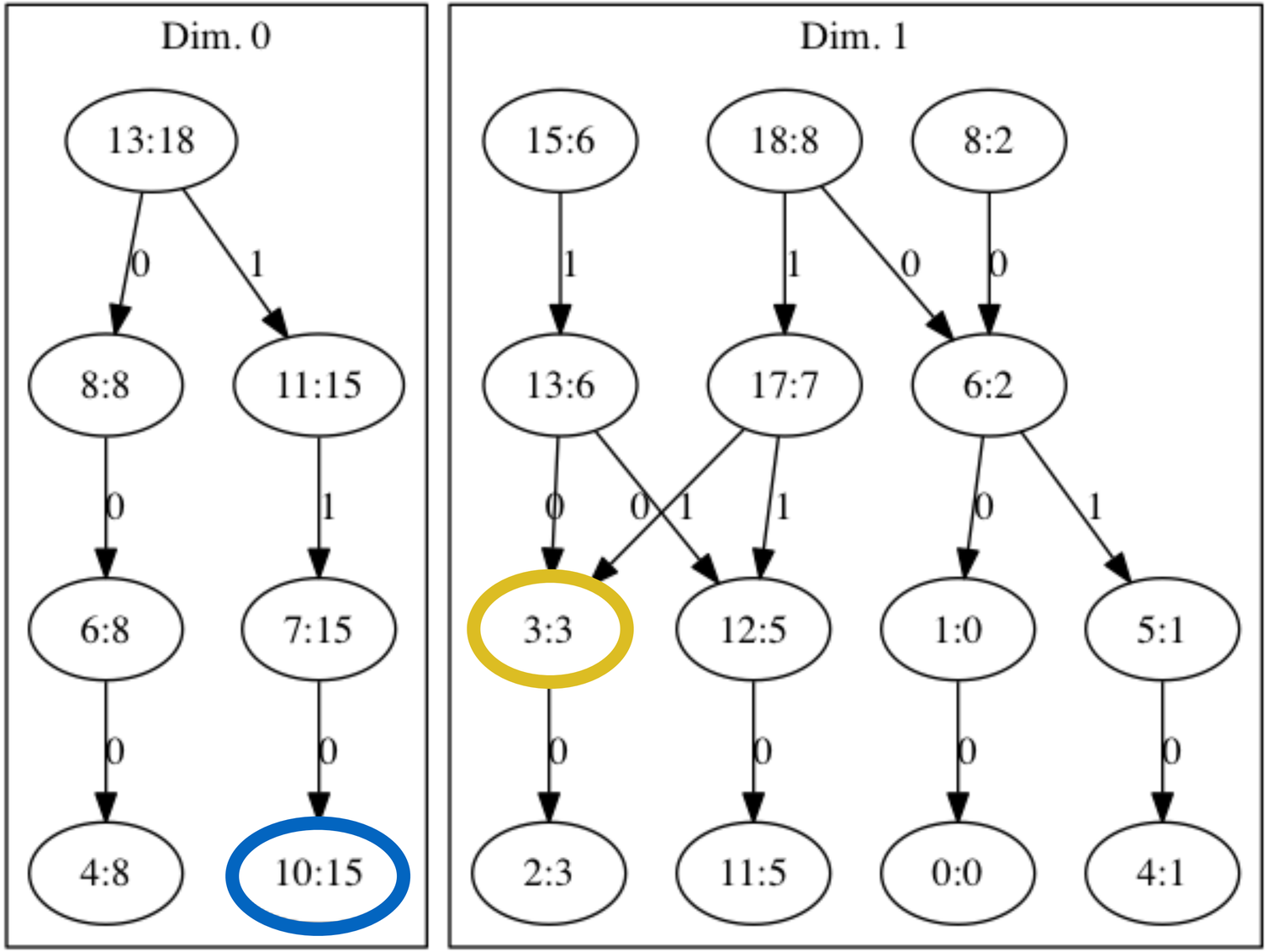
(000, 000)
(000, 010)
(110, 100)
(110, 110)



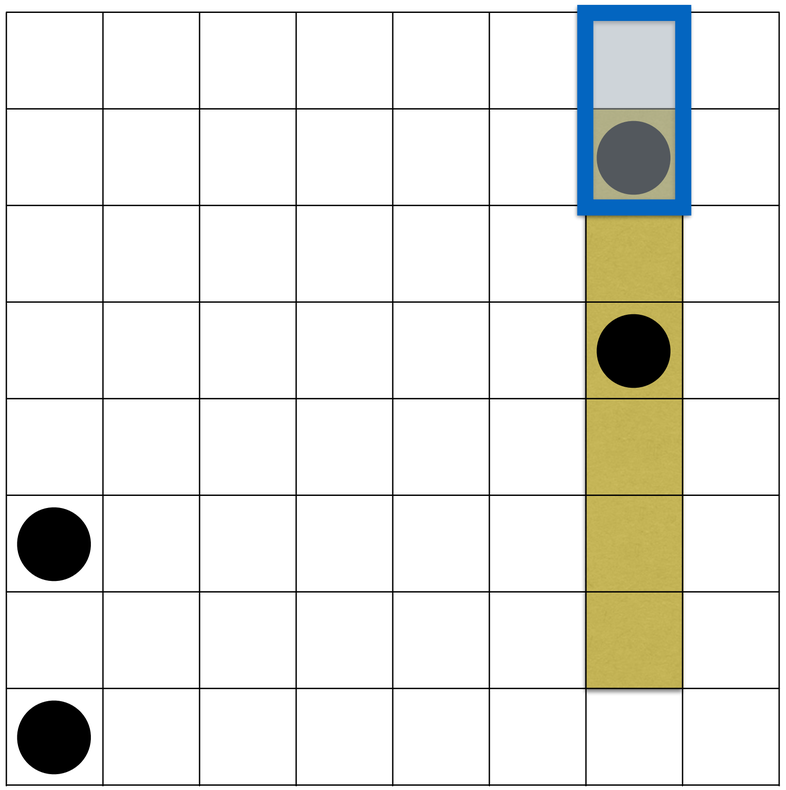
Example 2



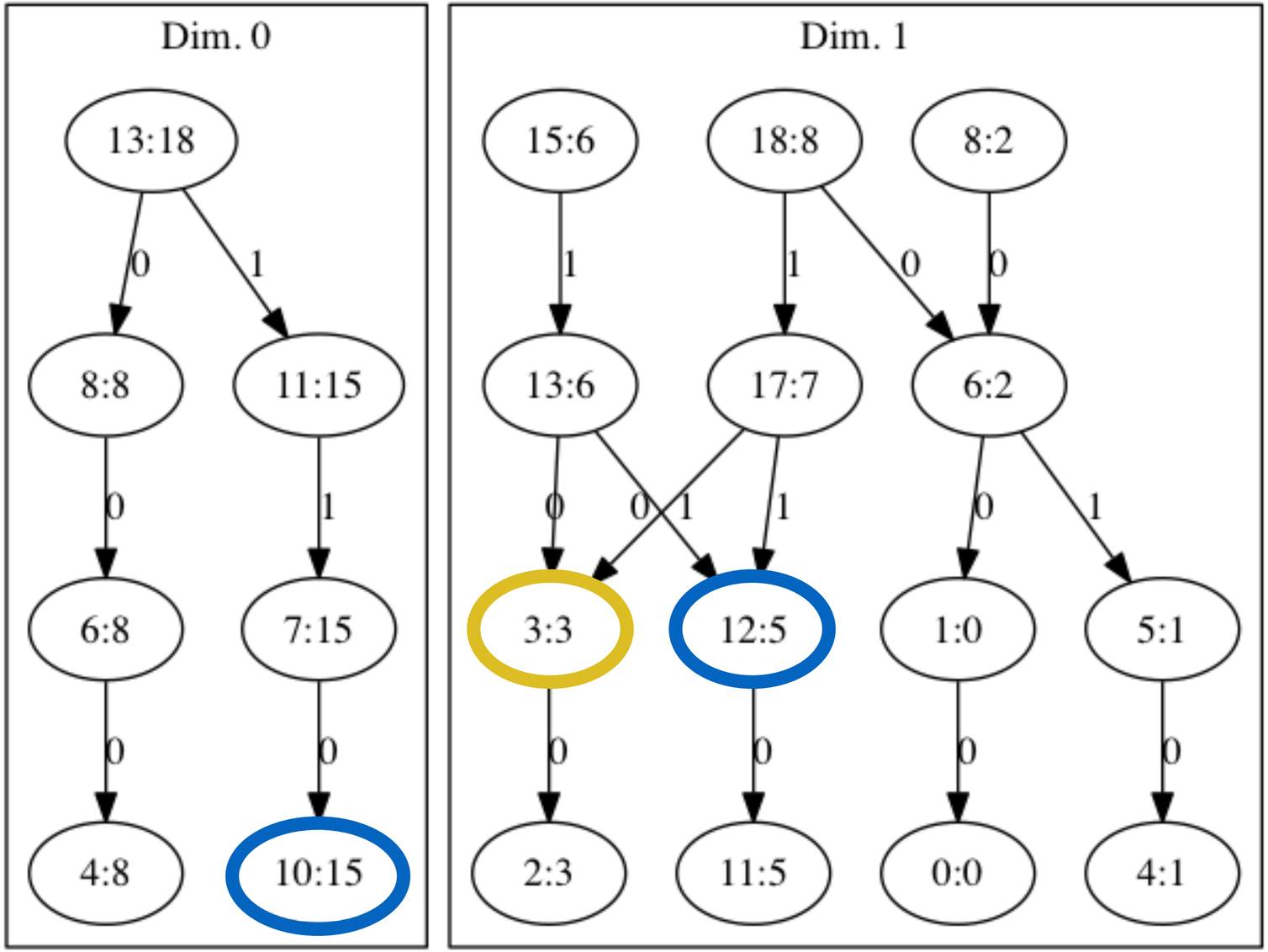
(000, 000)
(000, 010)
(110, 100)
(110, 110)



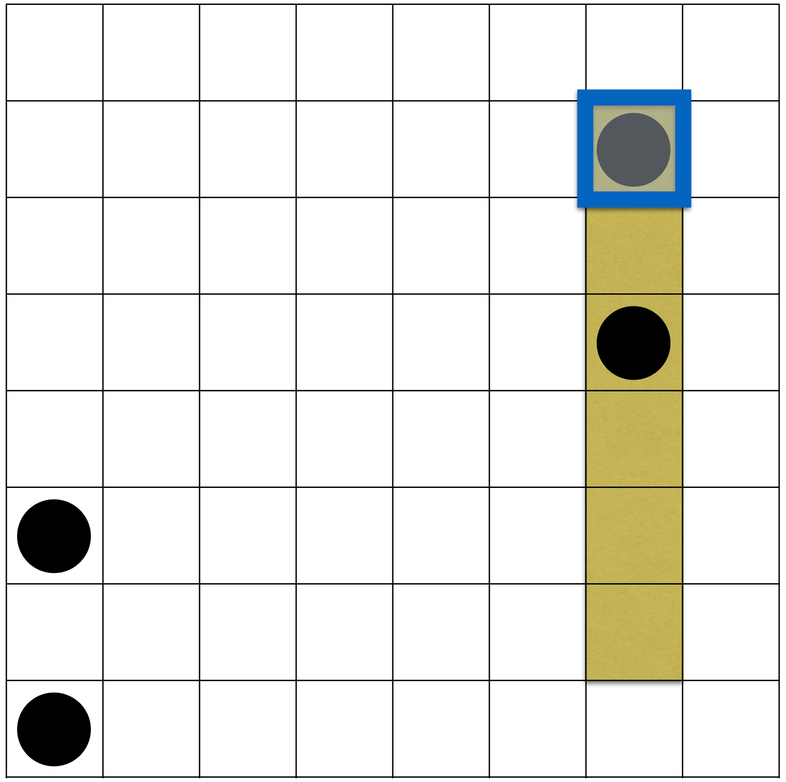
Example 2



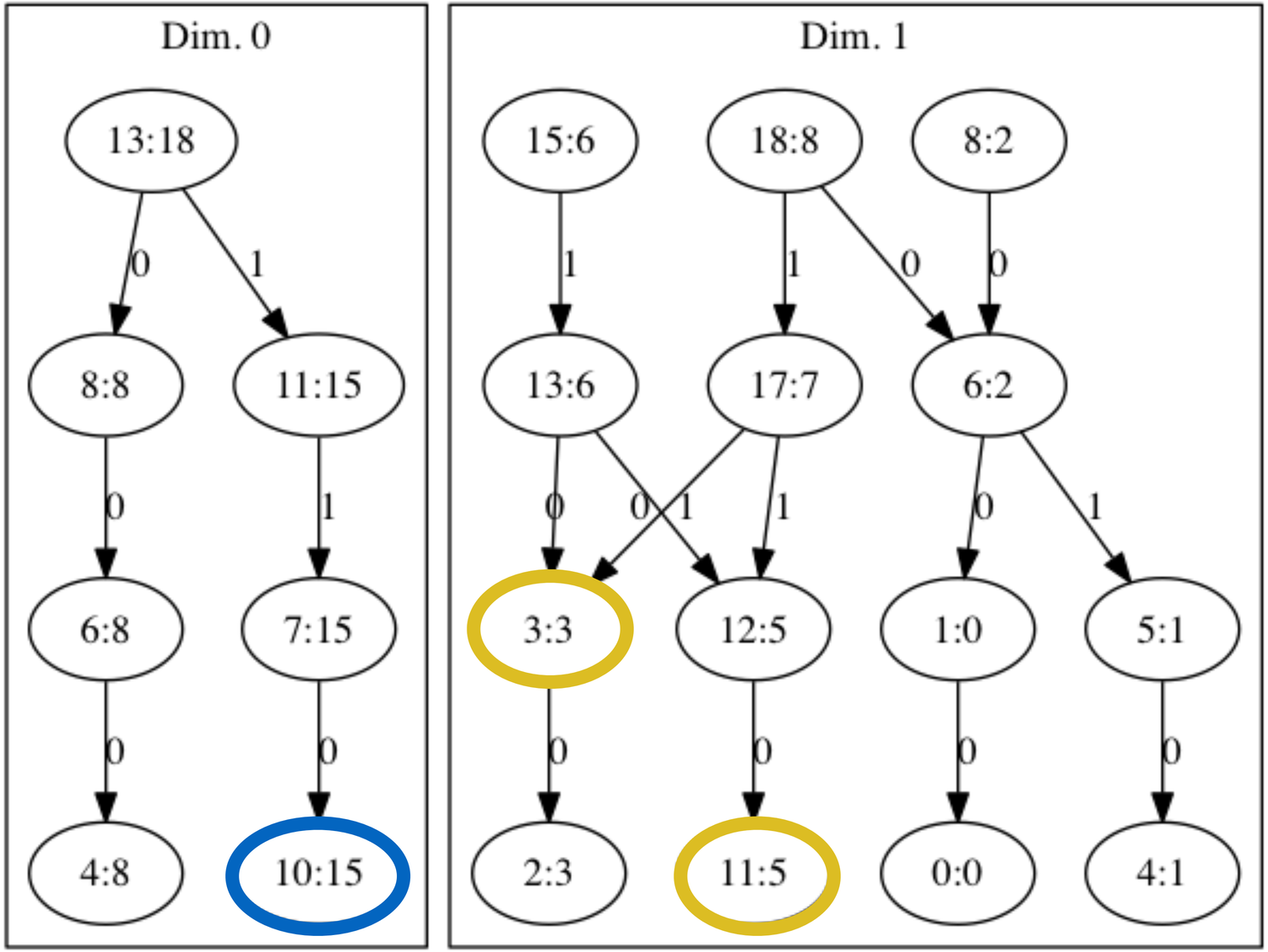
(000, 000)
(000, 010)
(110, 100)
(110, 110)



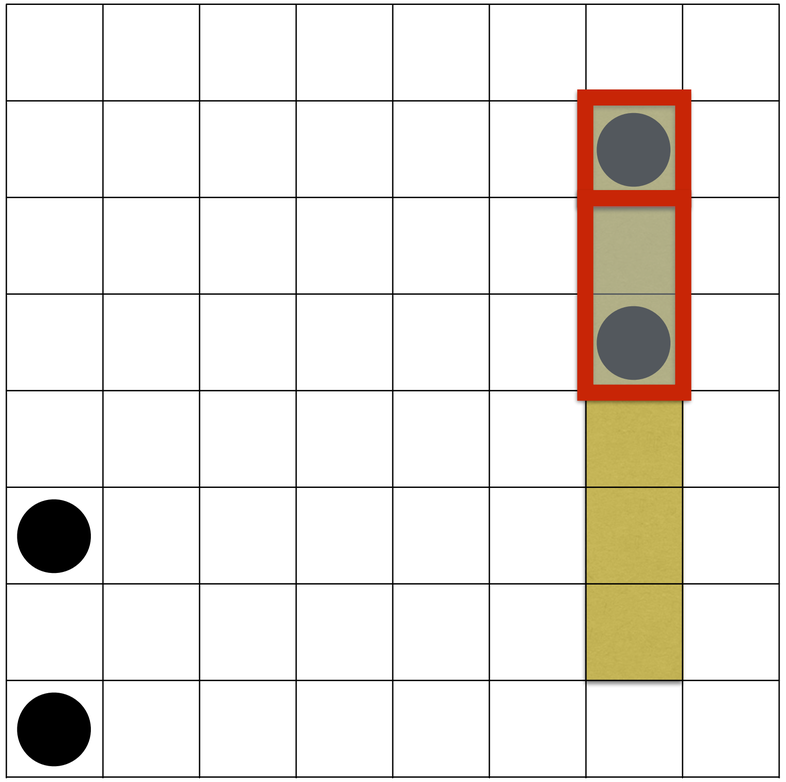
Example 2



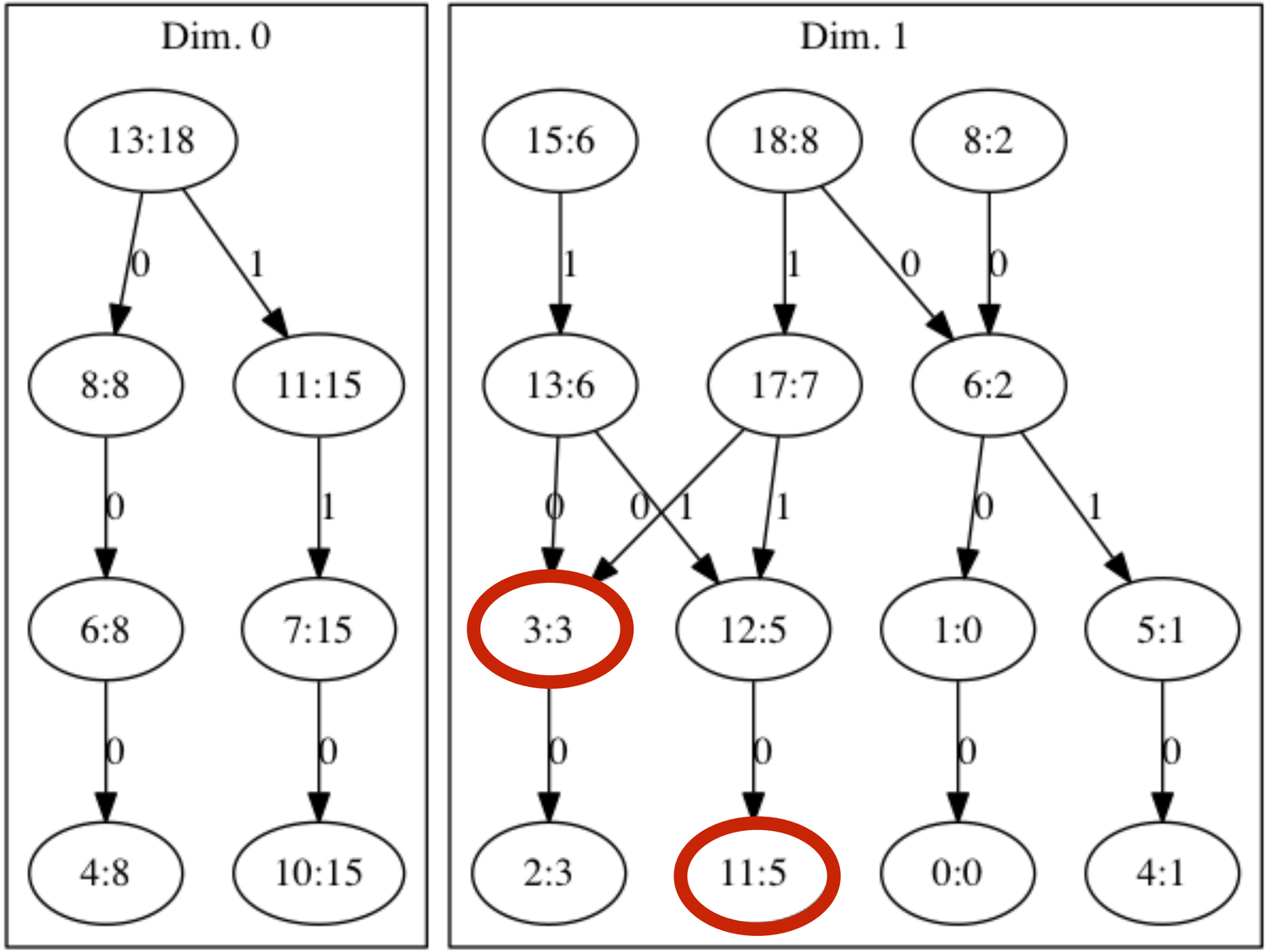
(000, 000)
(000, 010)
(110, 100)
(110, 110)



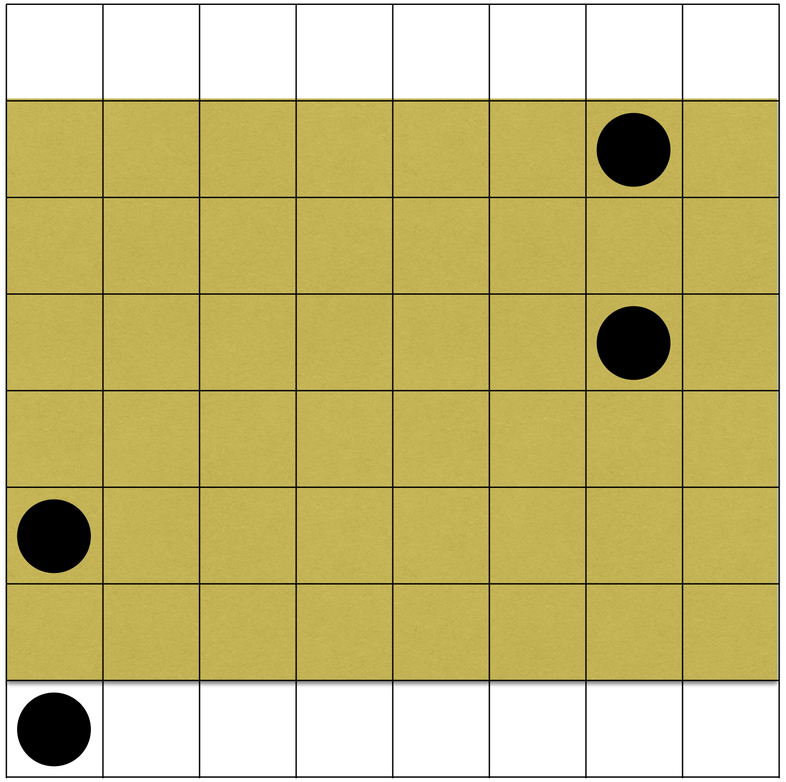
Example 2



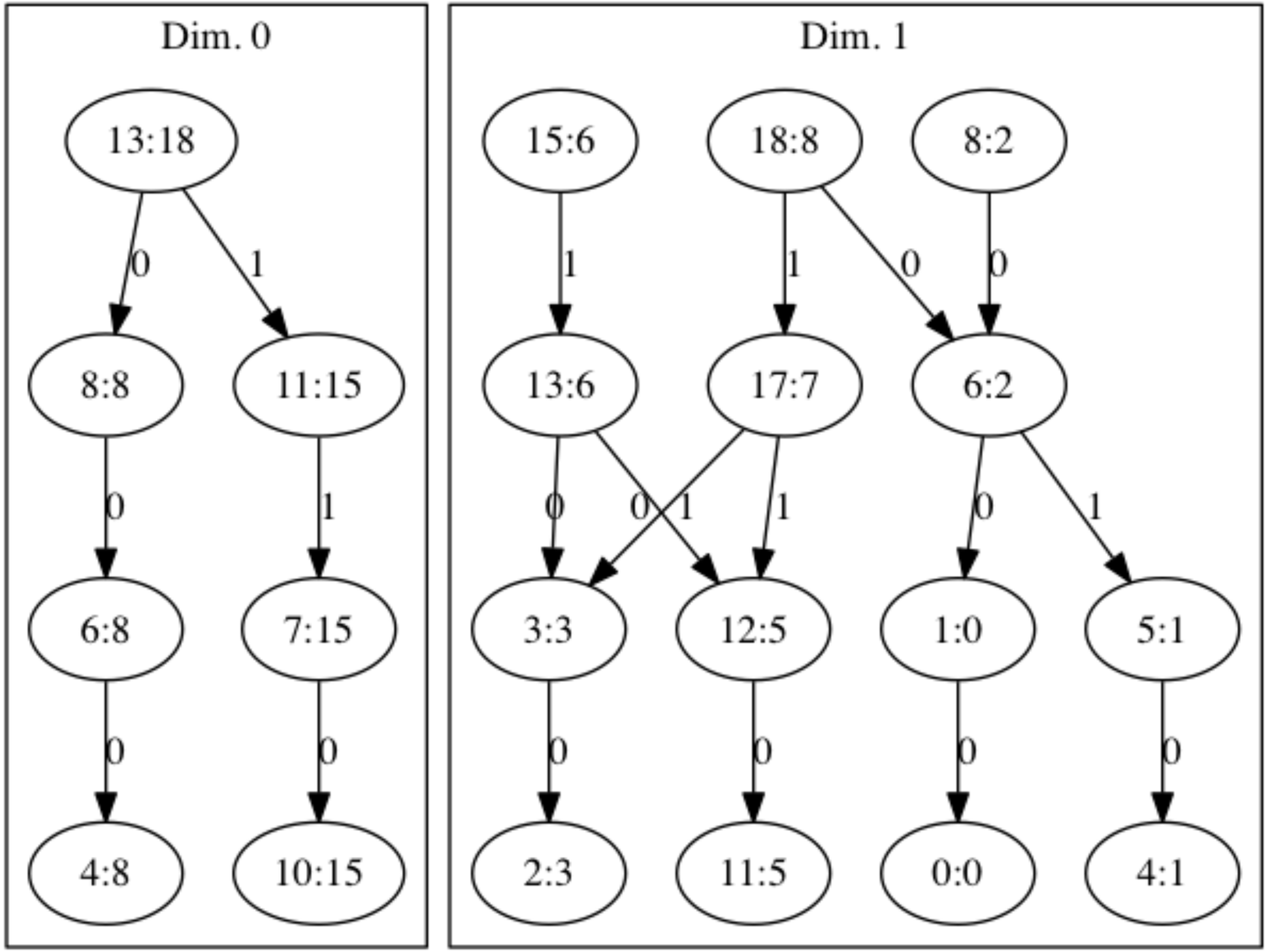
(000, 000)
(000, 010)
(110, 100)
(110, 110)



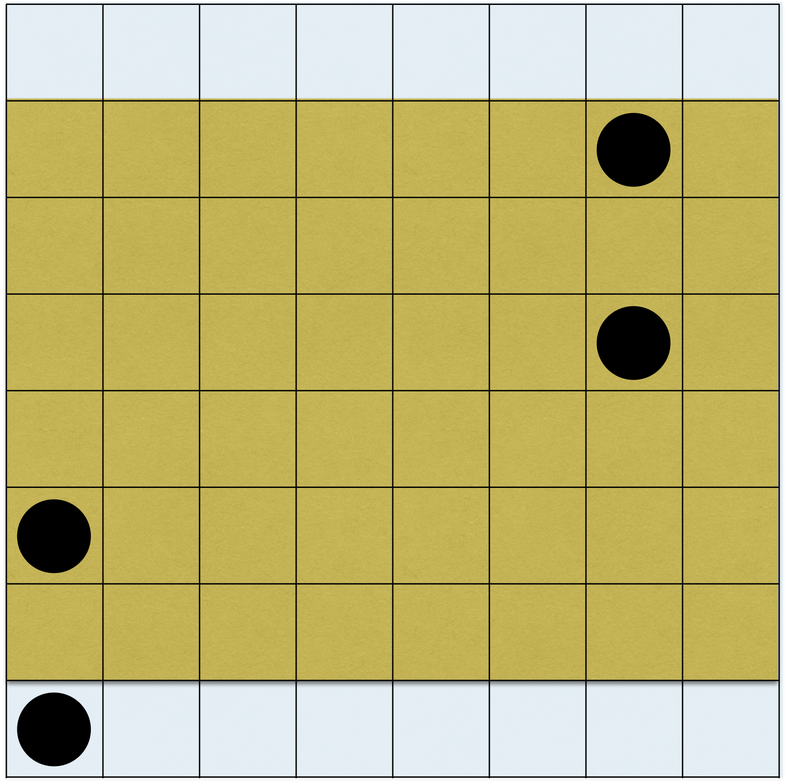
Example 3



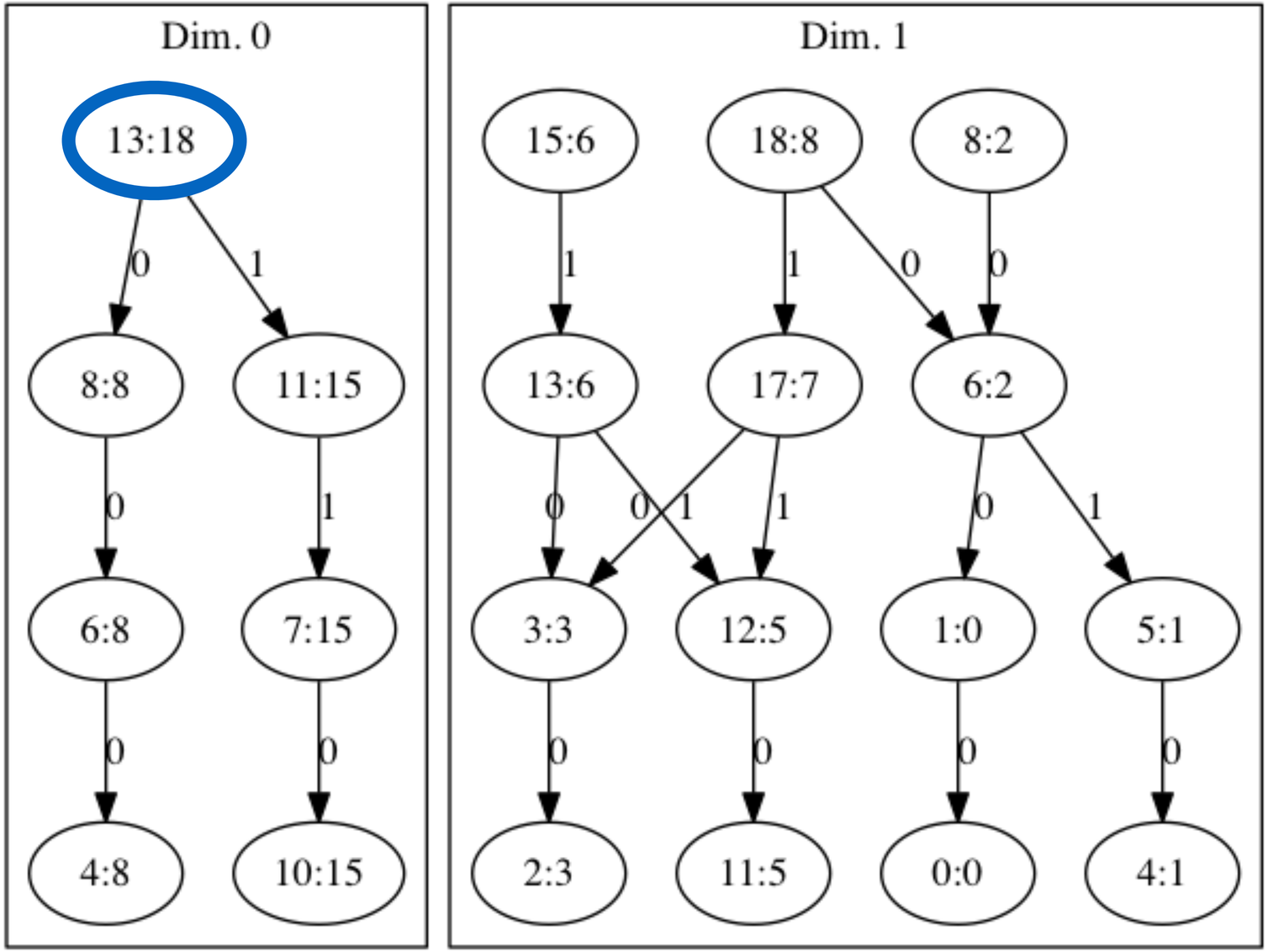
(000, 000)
(000, 010)
(110, 100)
(110, 110)



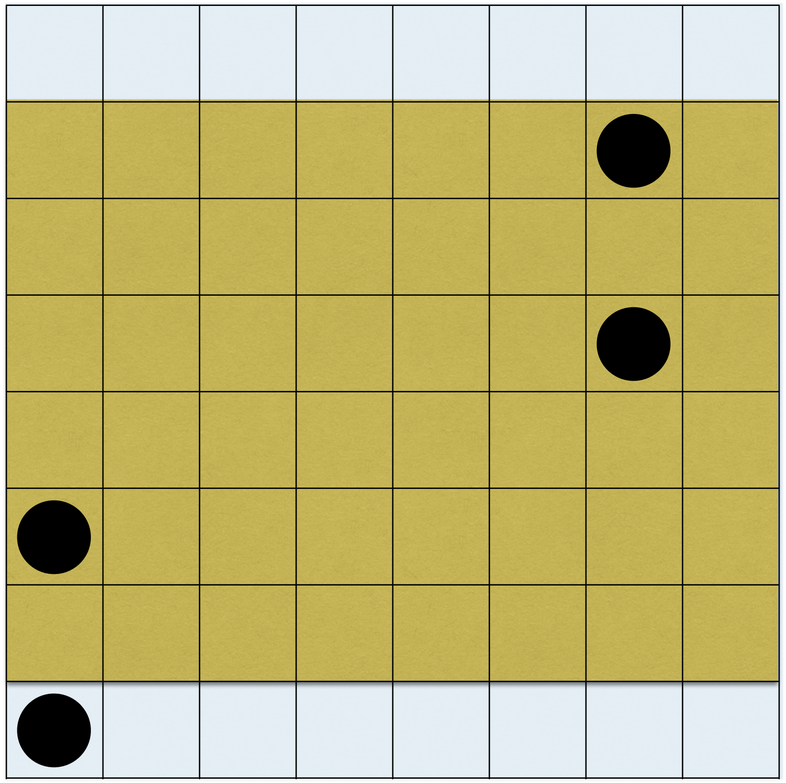
Example 3



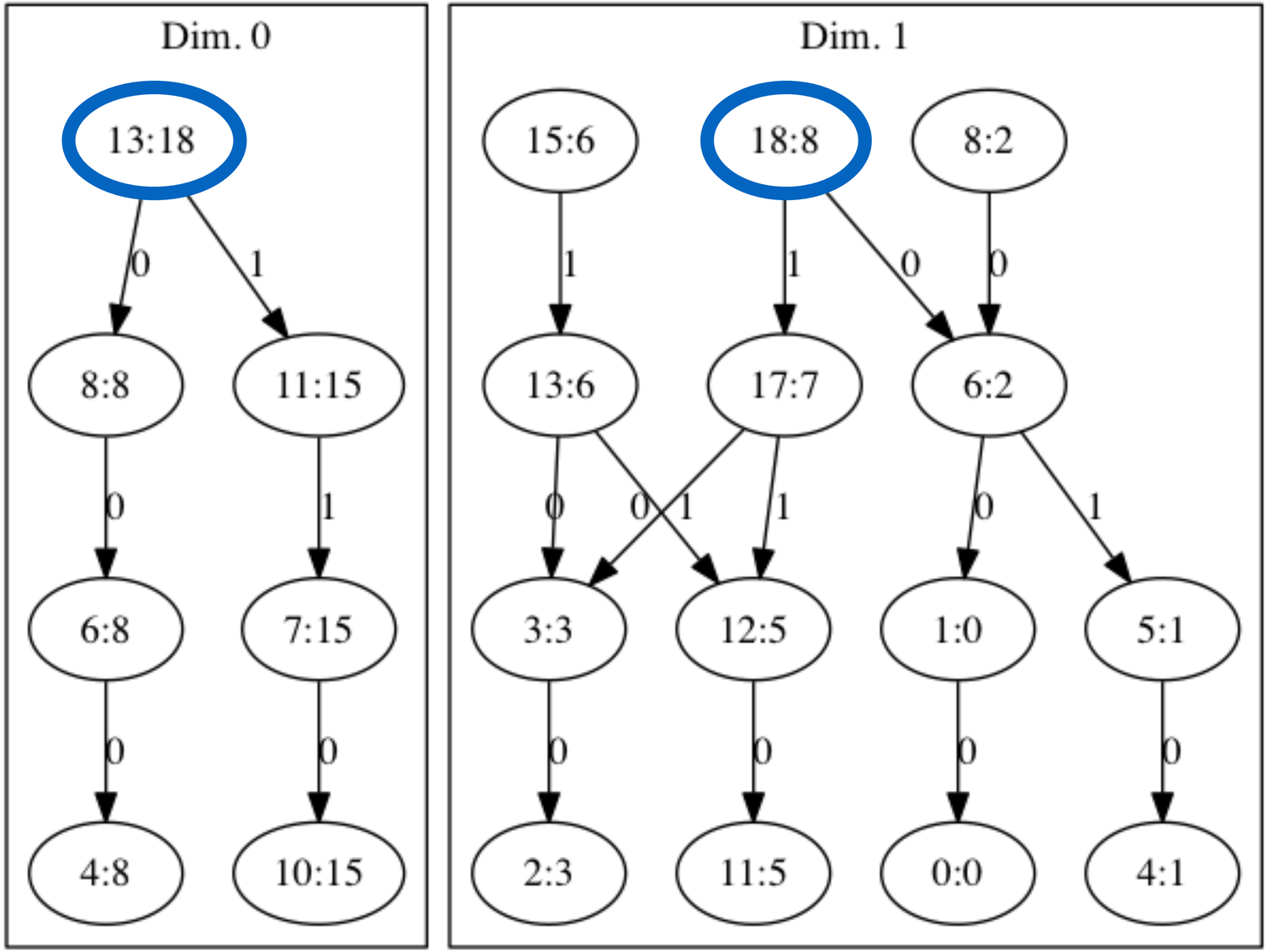
(000, 000)
(000, 010)
(110, 100)
(110, 110)



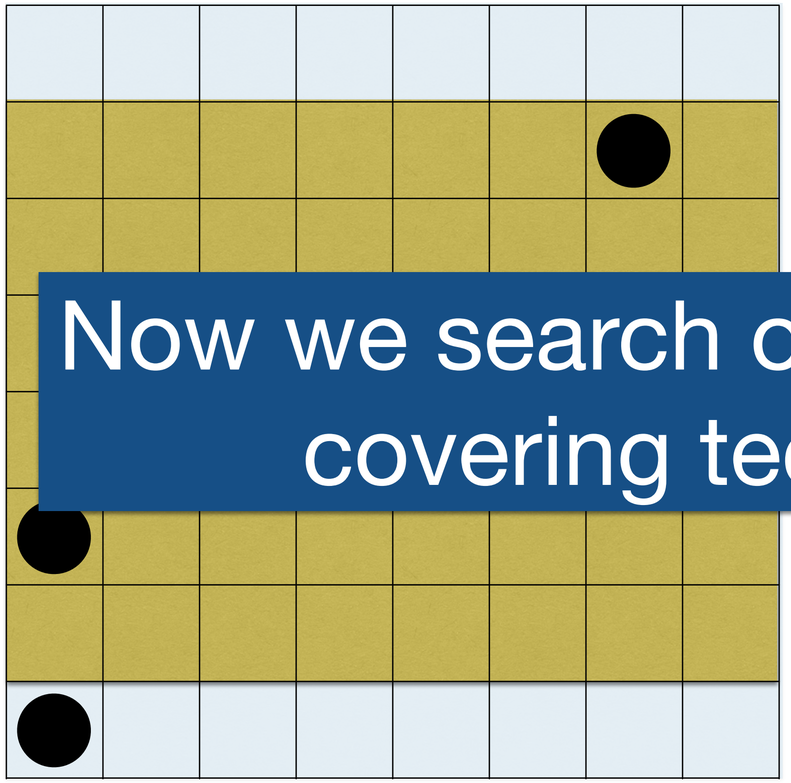
Example 3



(000, 000)
(000, 010)
(110, 100)
(110, 110)



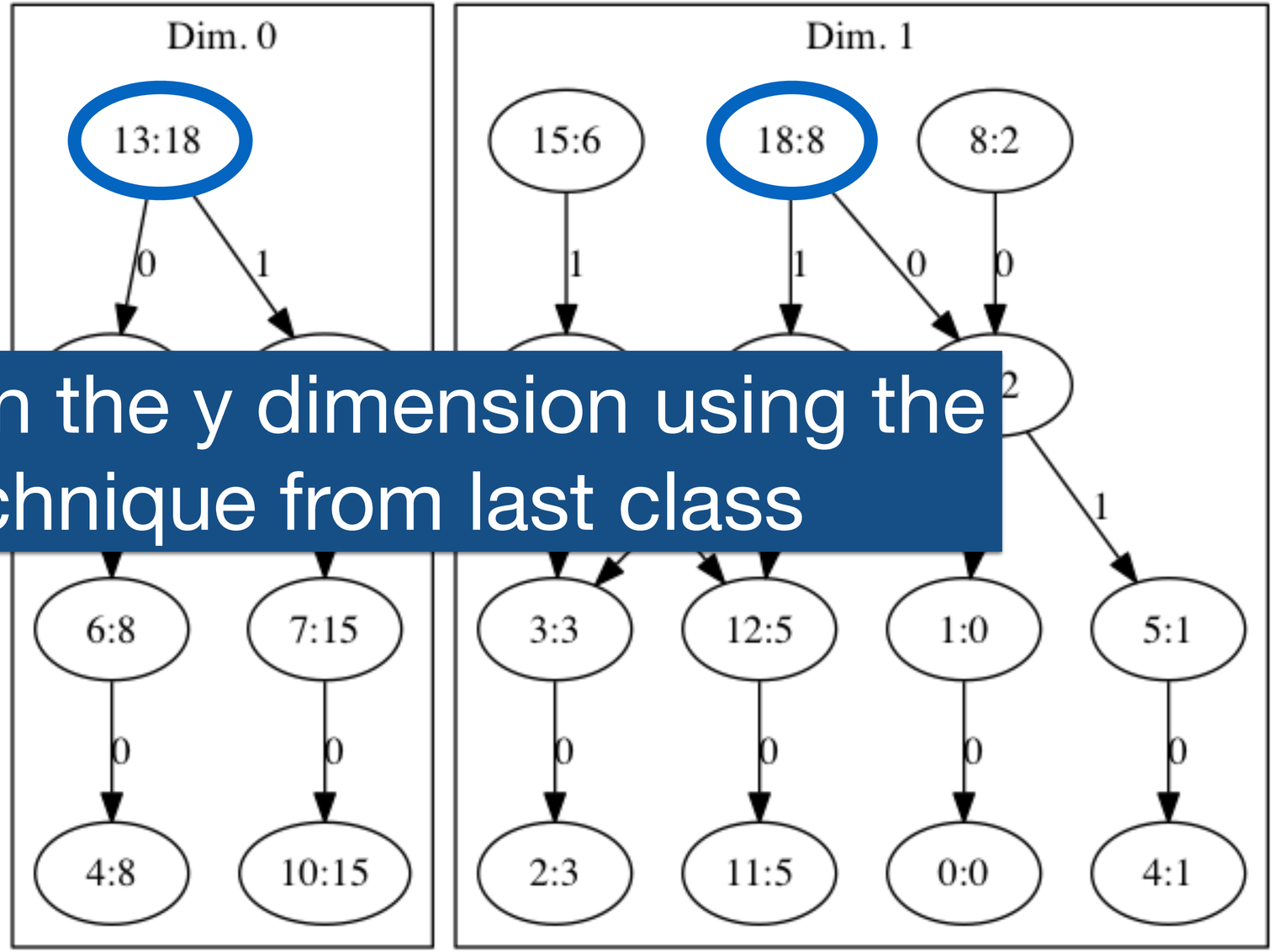
Example 3



y

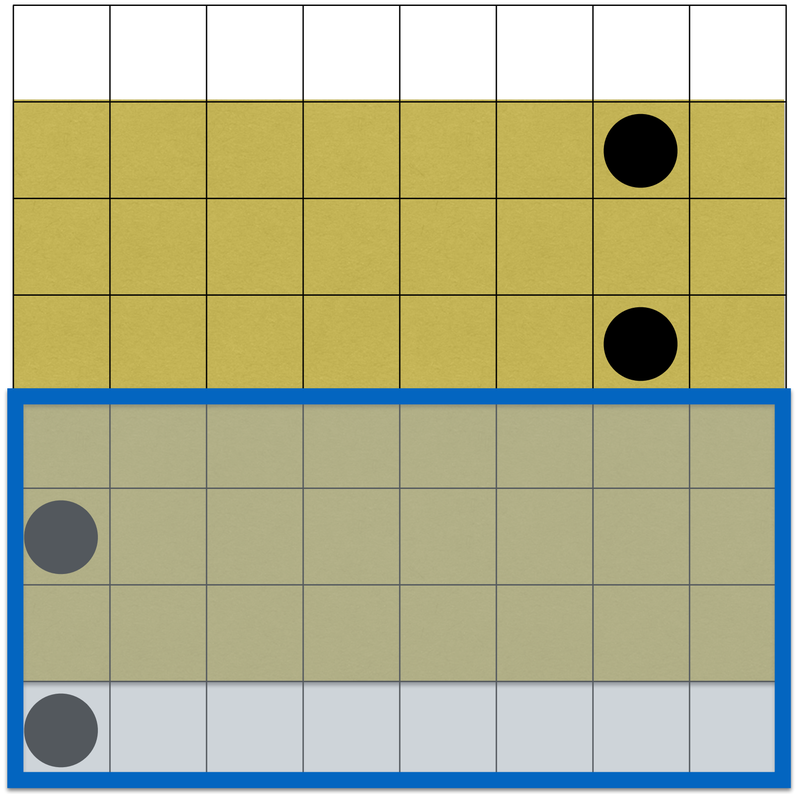
x

(000, 000)
(000, 010)
(110, 100)
(110, 110)

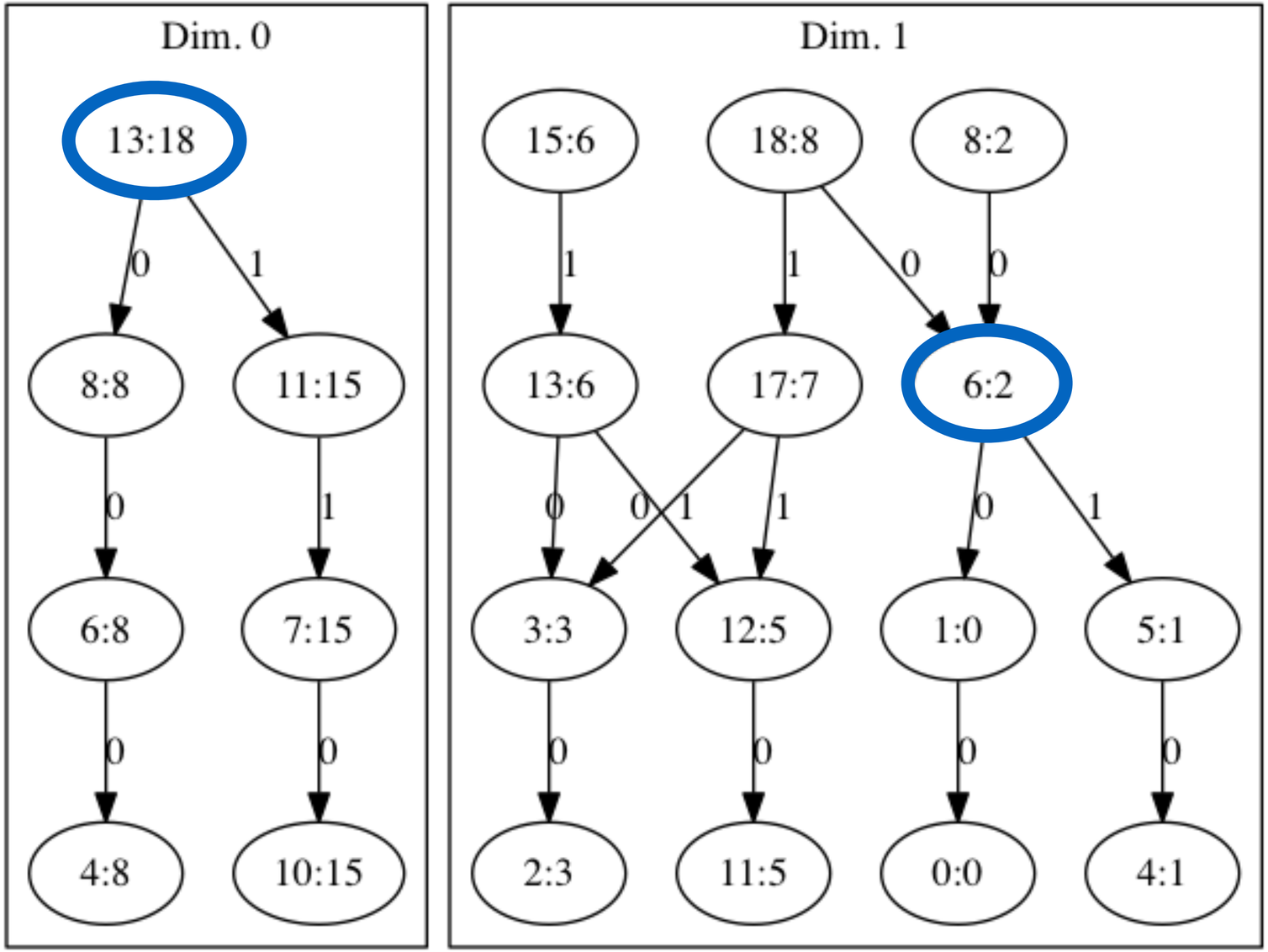


Now we search on the y dimension using the covering technique from last class

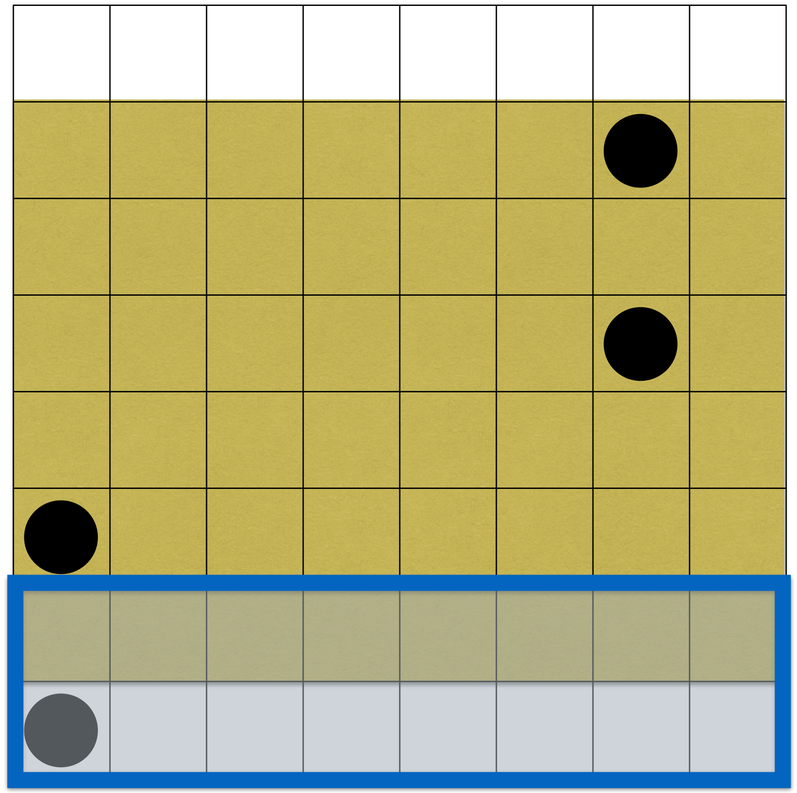
Example 3



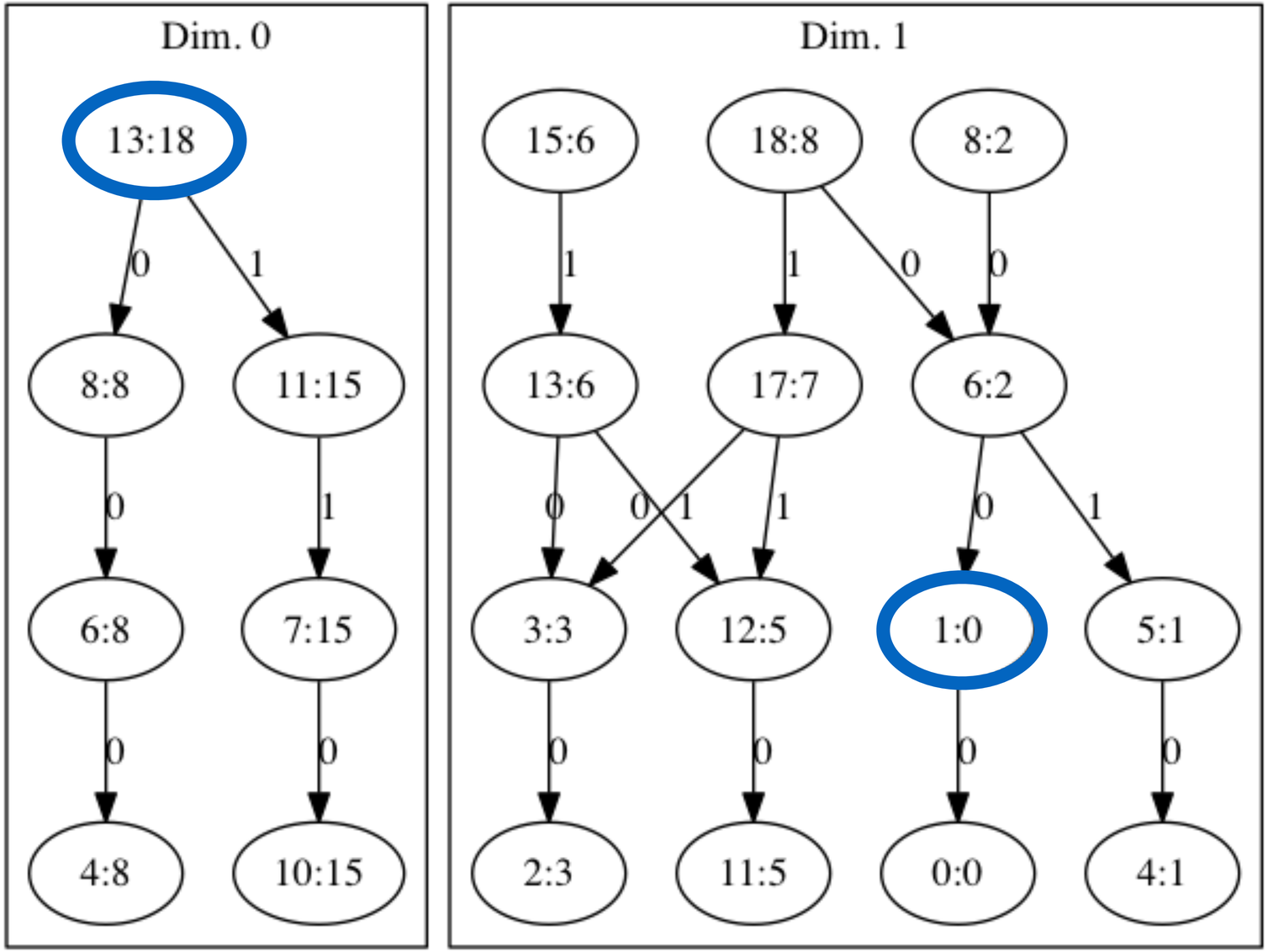
(000, 000)
(000, 010)
(110, 100)
(110, 110)



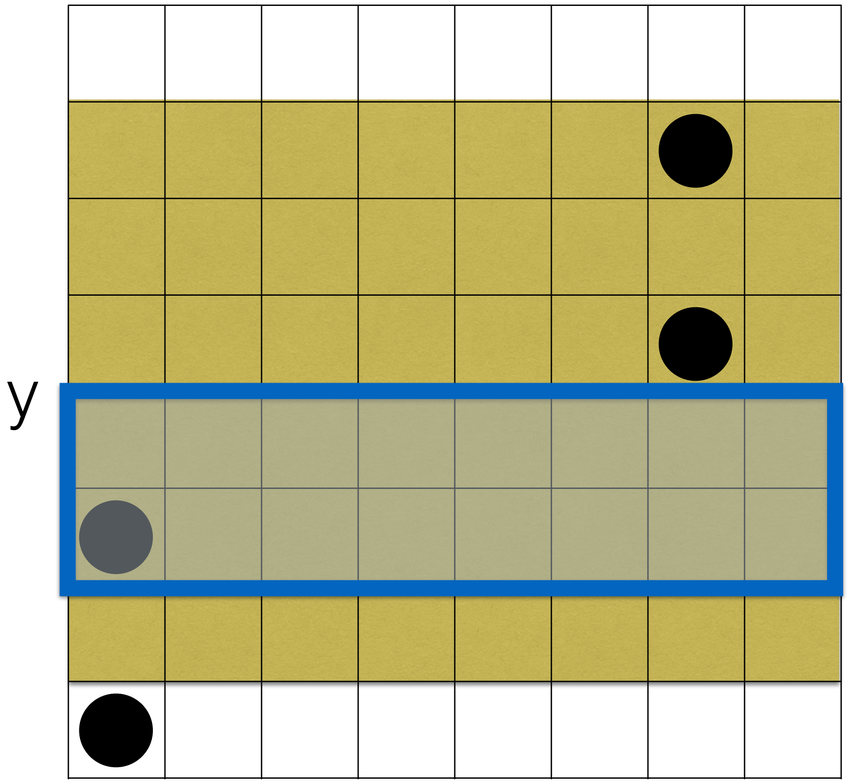
Example 3



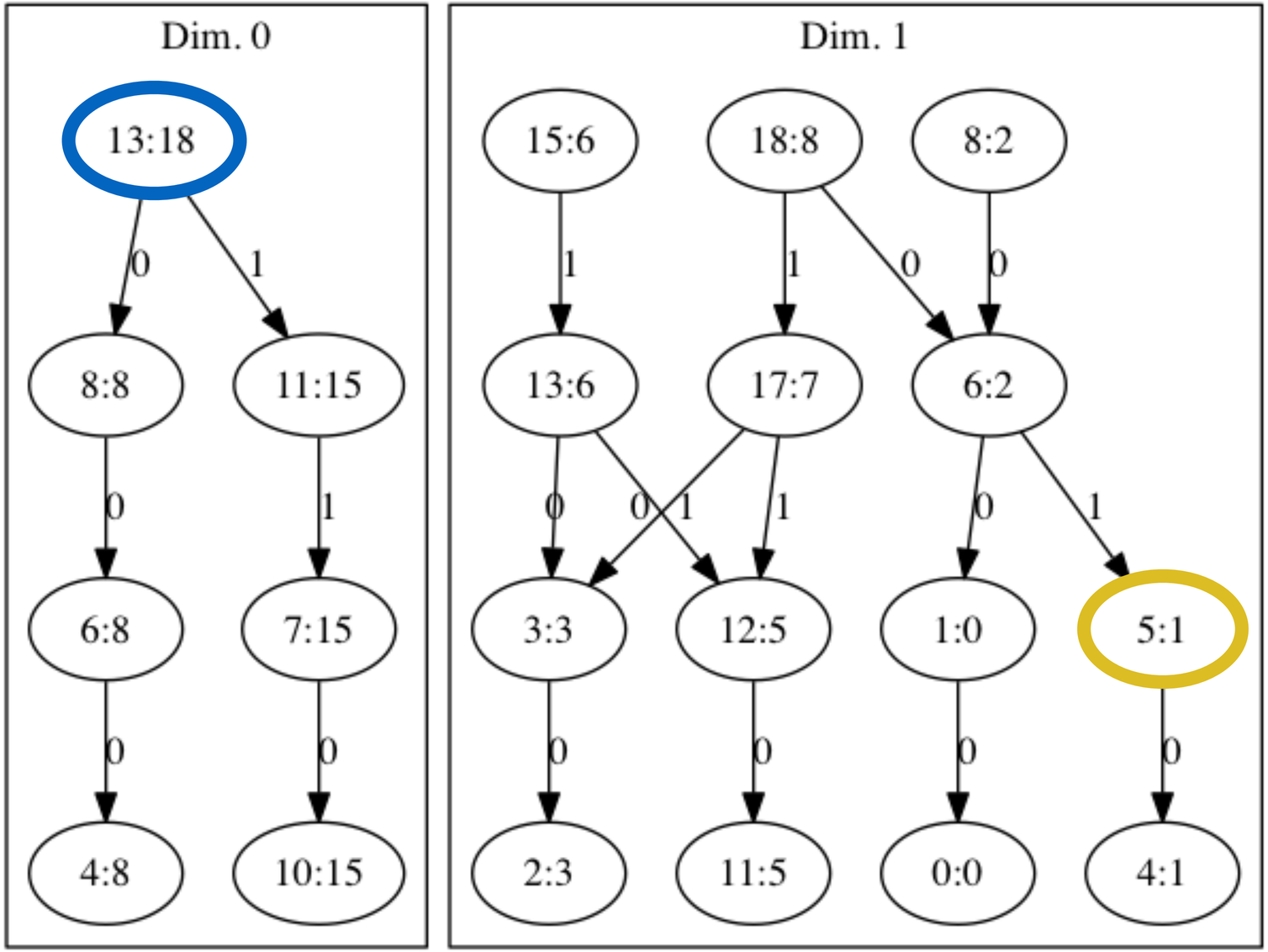
(000, 000)
(000, 010)
(110, 100)
(110, 110)



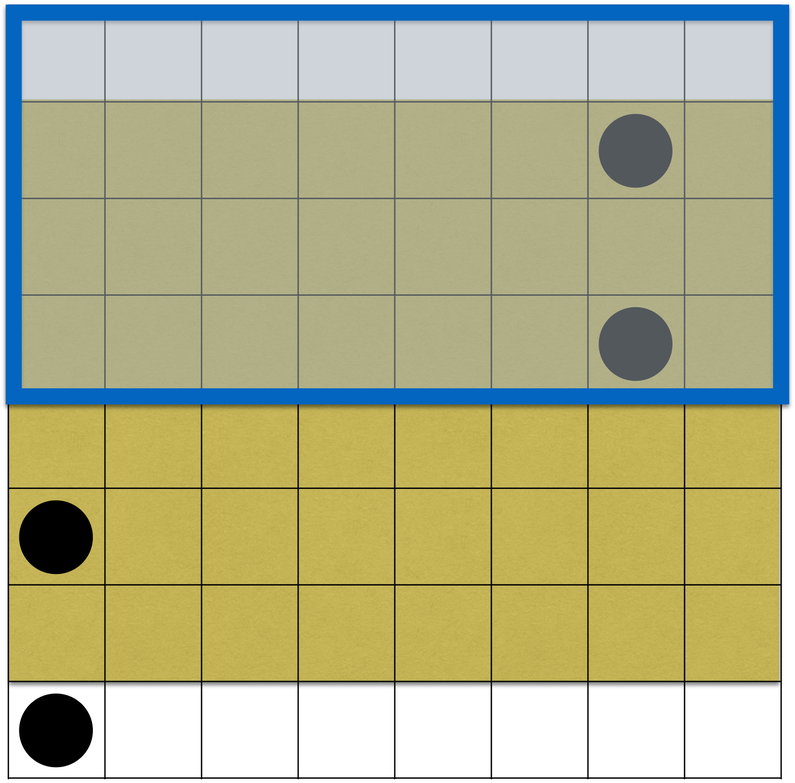
Example 3



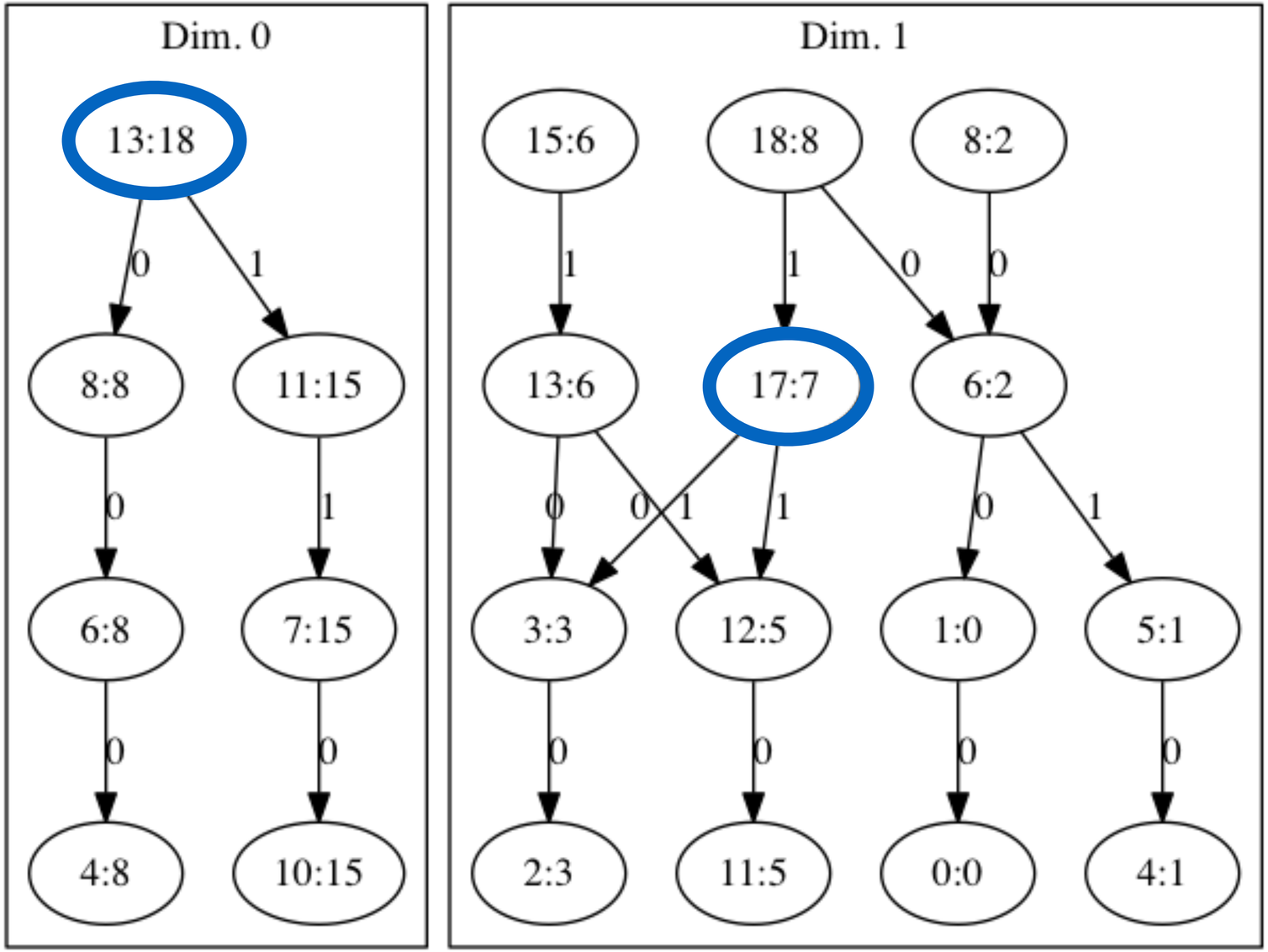
(000, 000)
(000, 010)
(110, 100)
(110, 110)



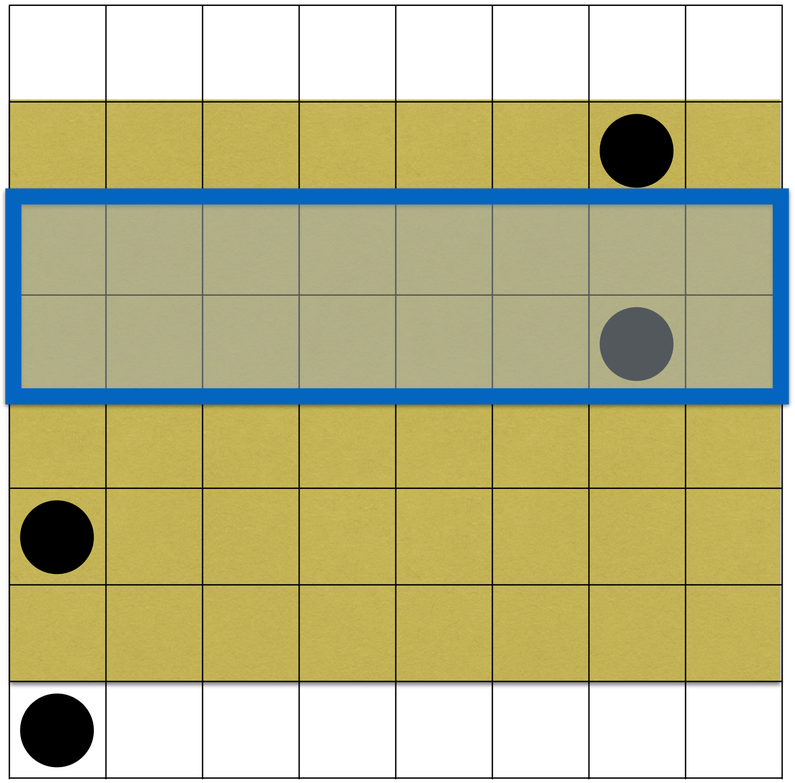
Example 3



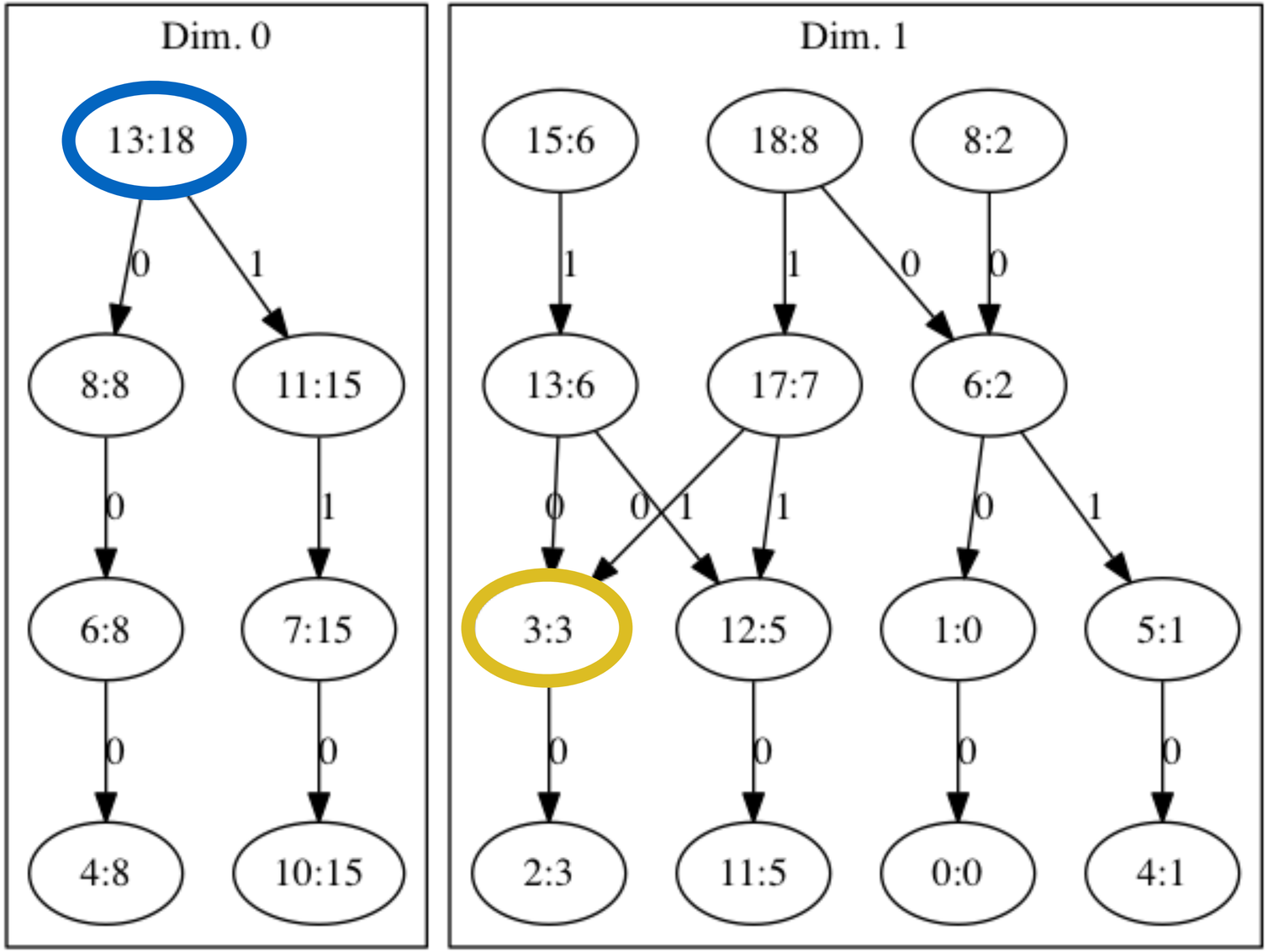
(000, 000)
(000, 010)
(110, 100)
(110, 110)



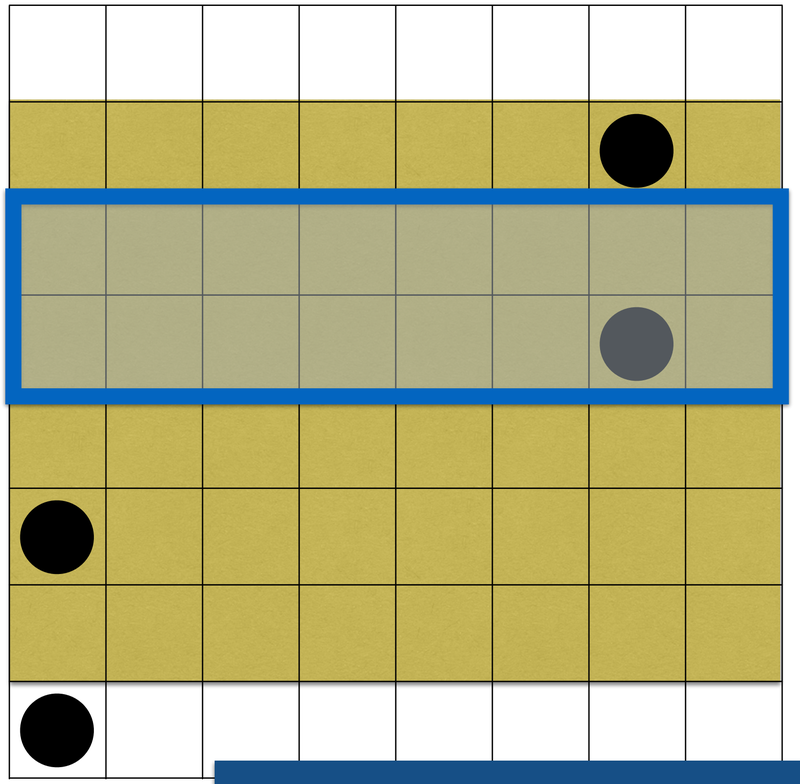
Example 3

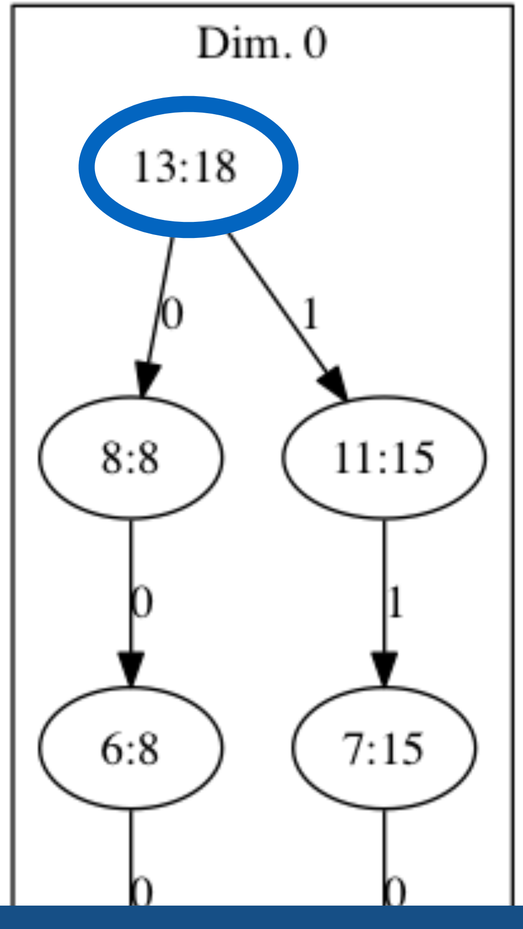


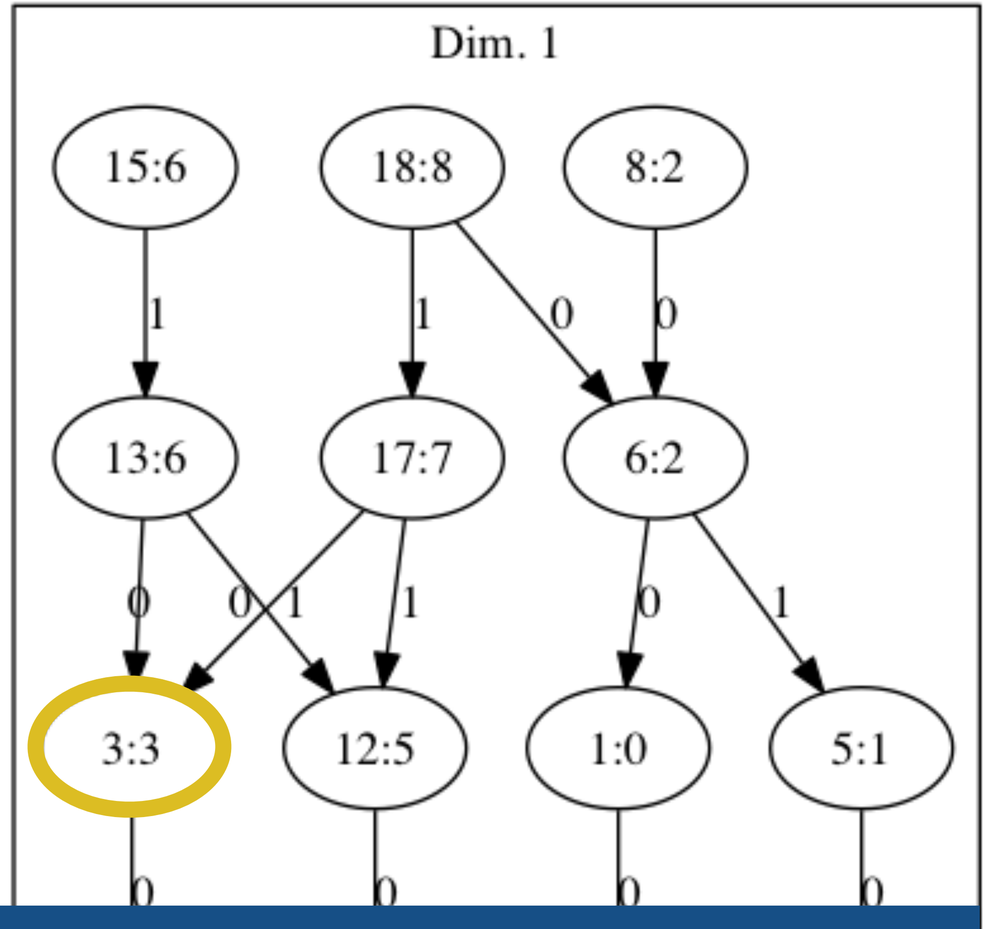
(000, 000)
(000, 010)
(110, 100)
(110, 110)



Example 3



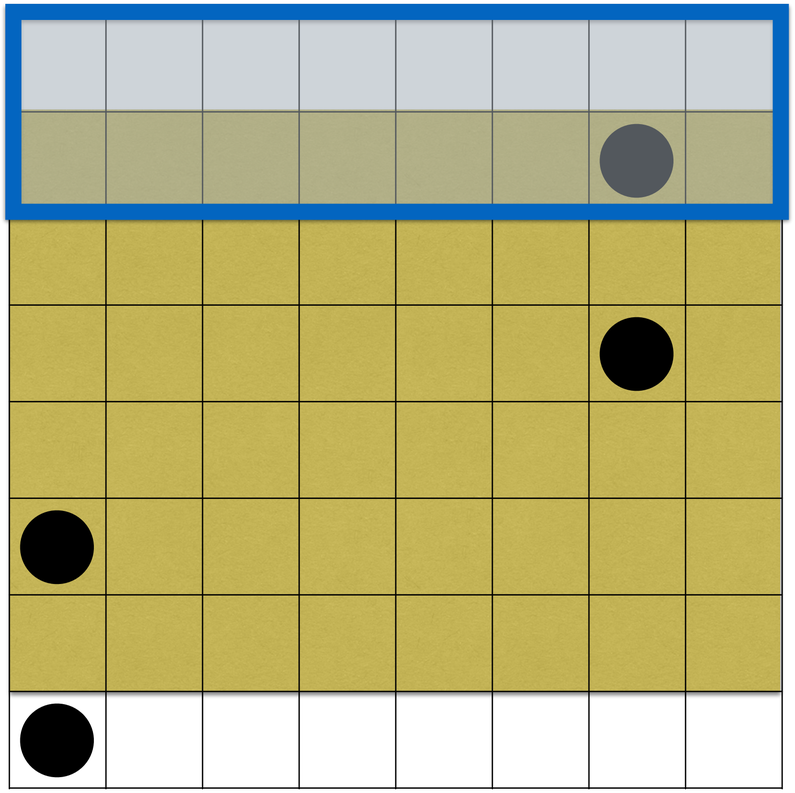




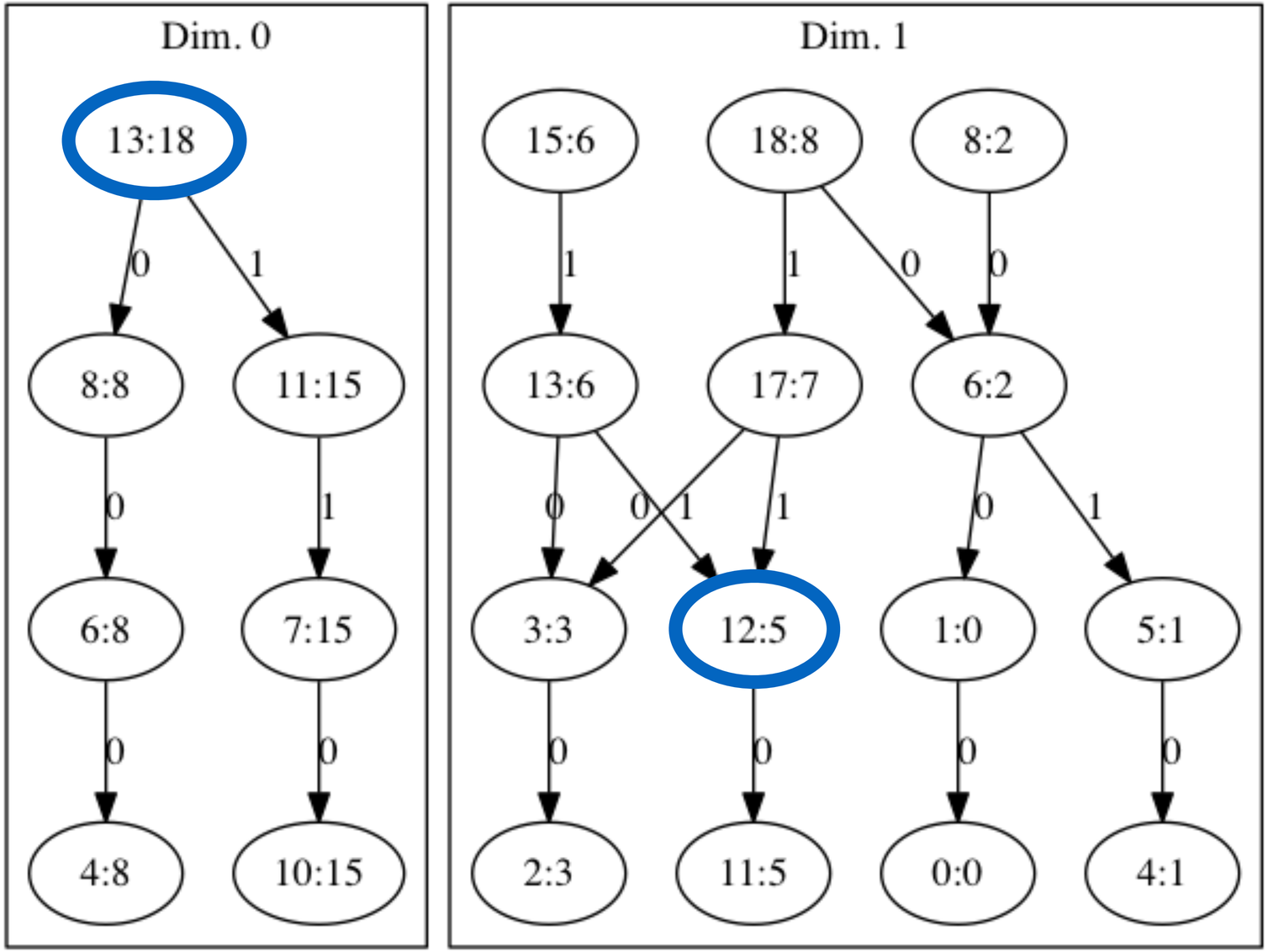
(000, 000)
(000, 010)
(110, 100)
(110, 110)

Note that we have visited this node before, on a different search! This is how shared pieces of the data structure allow space savings.

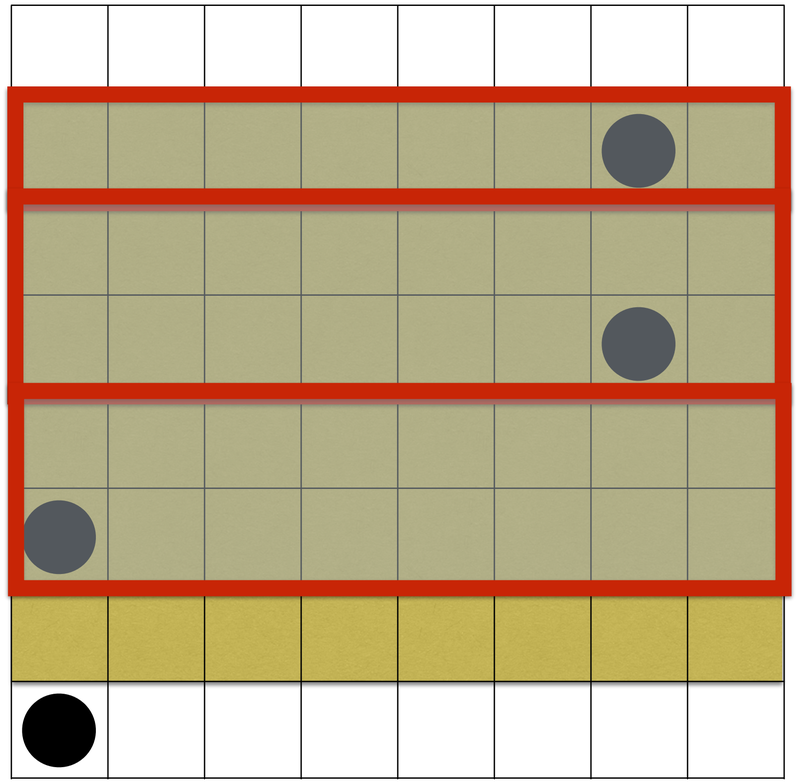
Example 3



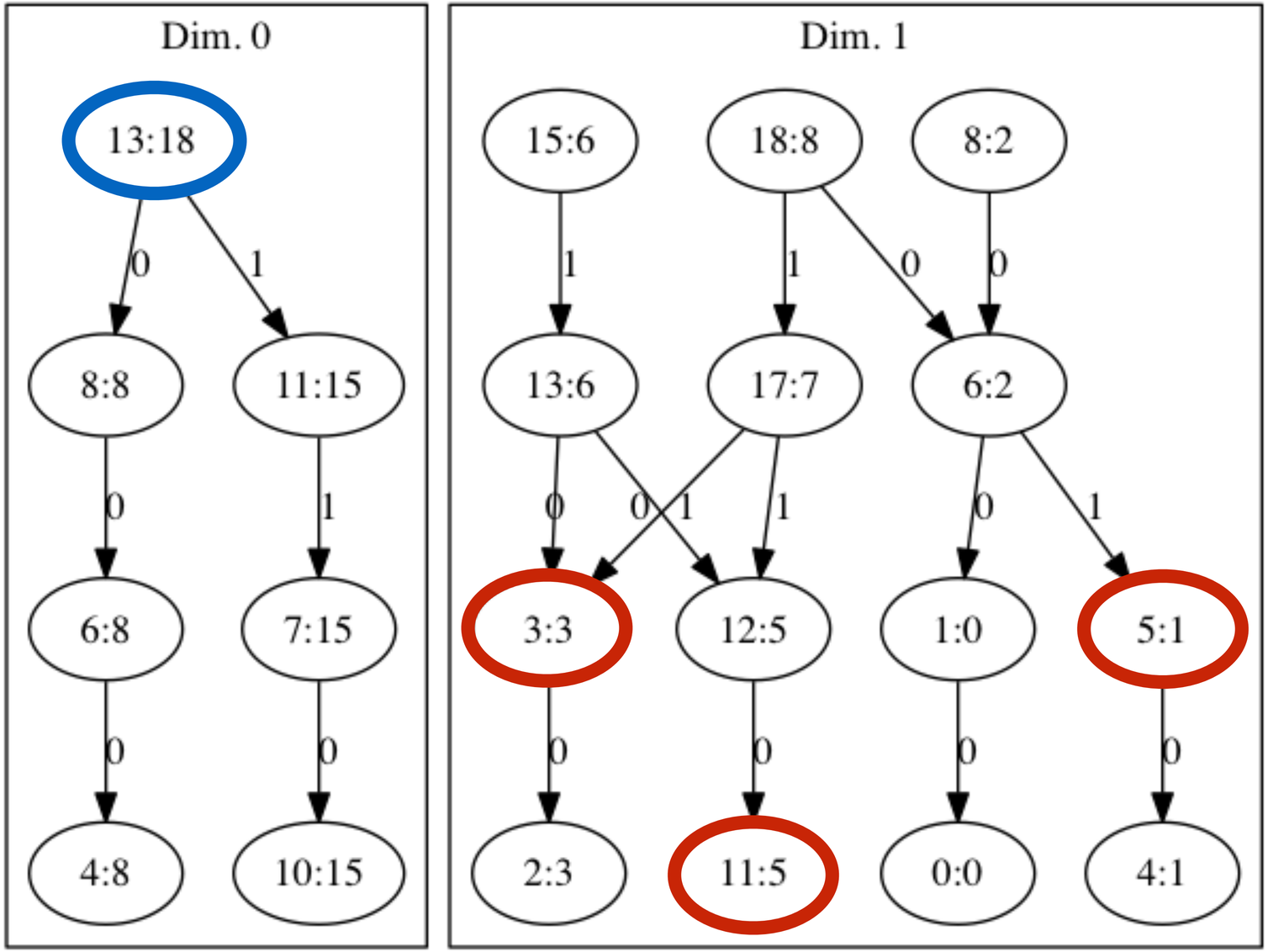
(000, 000)
(000, 010)
(110, 100)
(110, 110)



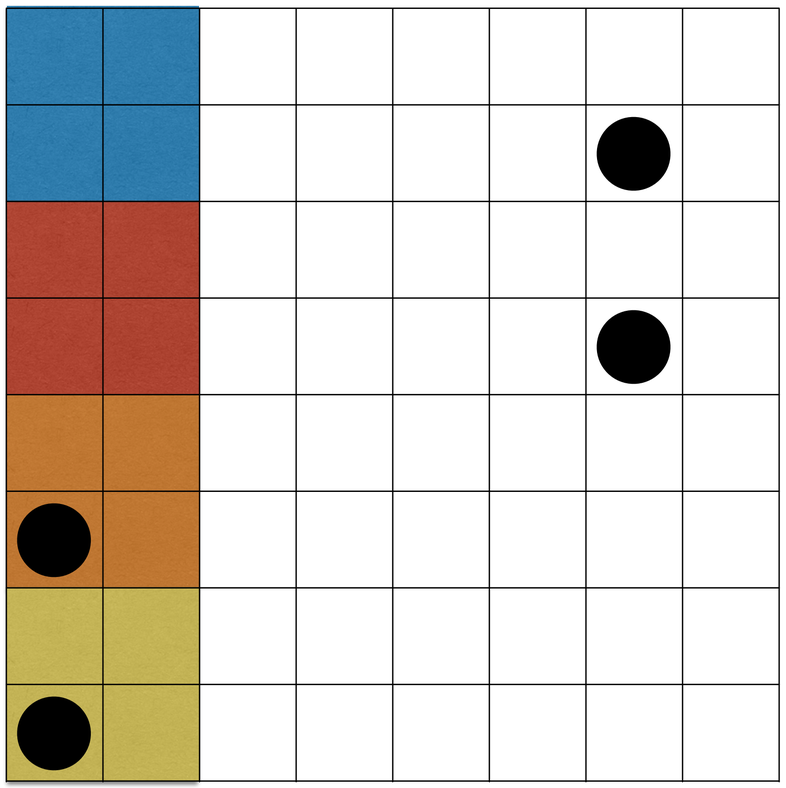
Example 3



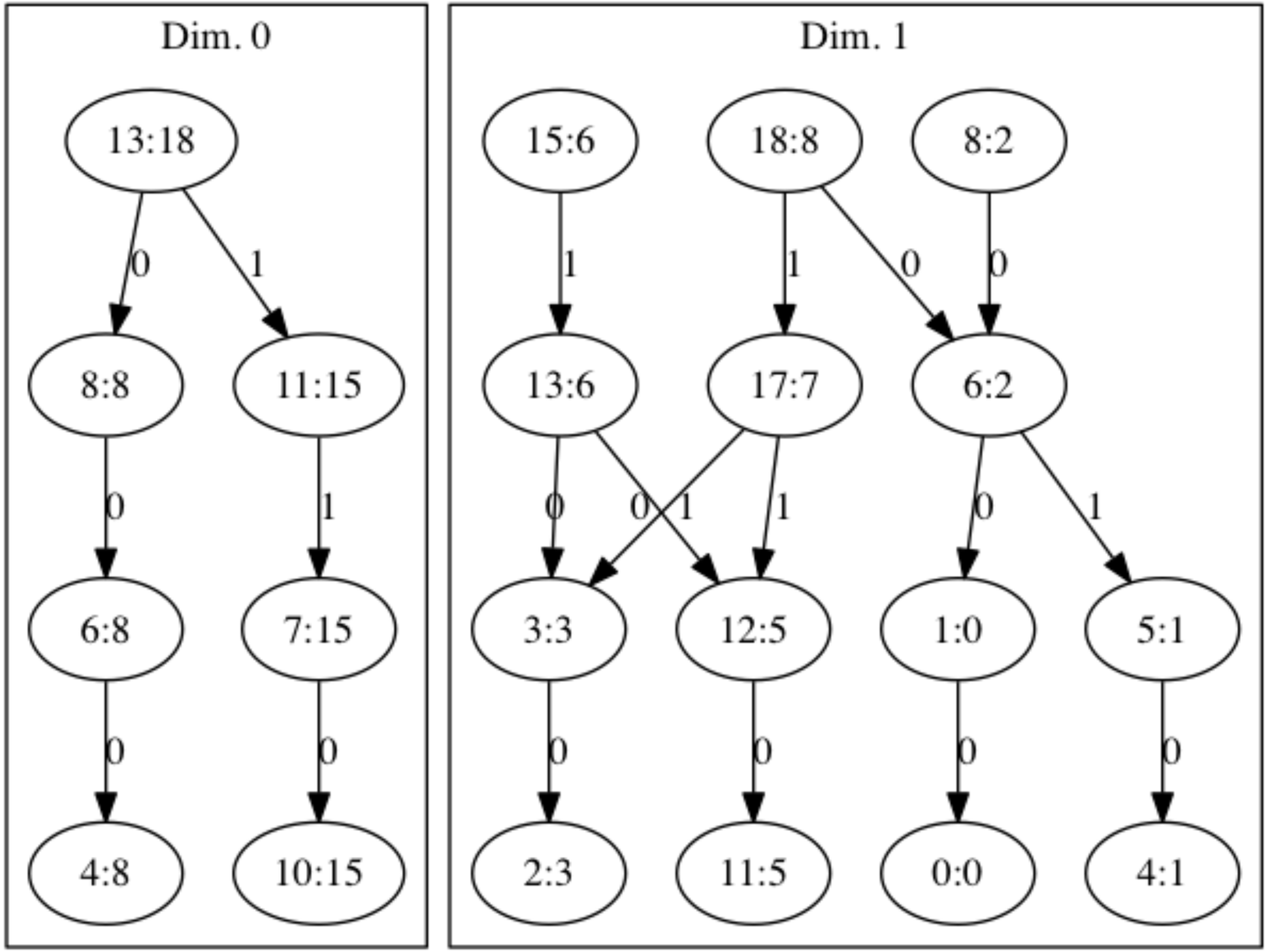
(000, 000)
(000, 010)
(110, 100)
(110, 110)



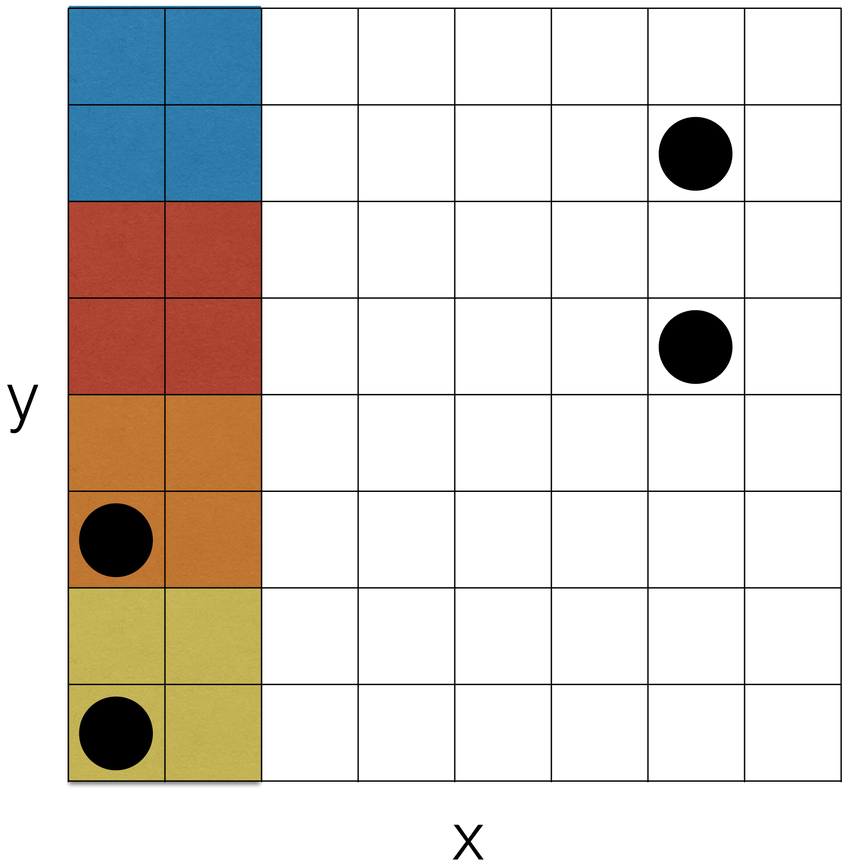
Example 4



(000, 000)
(000, 010)
(110, 100)
(110, 110)

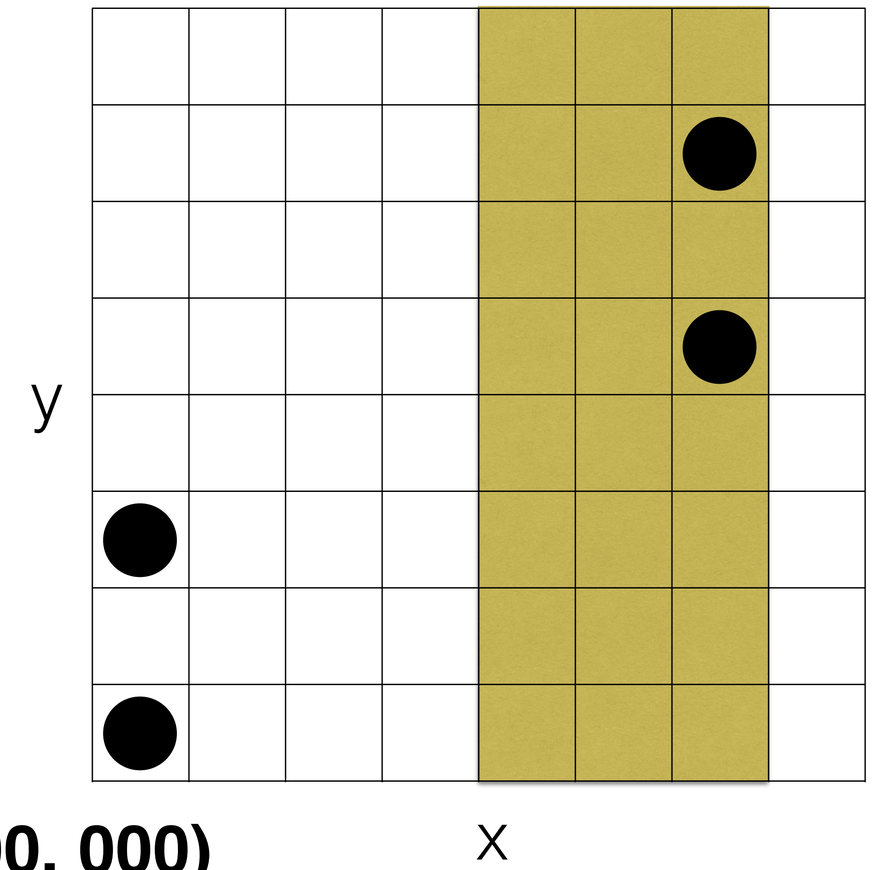


Example 4



What if we build a single dimension for both X and Y, like a quad-tree?

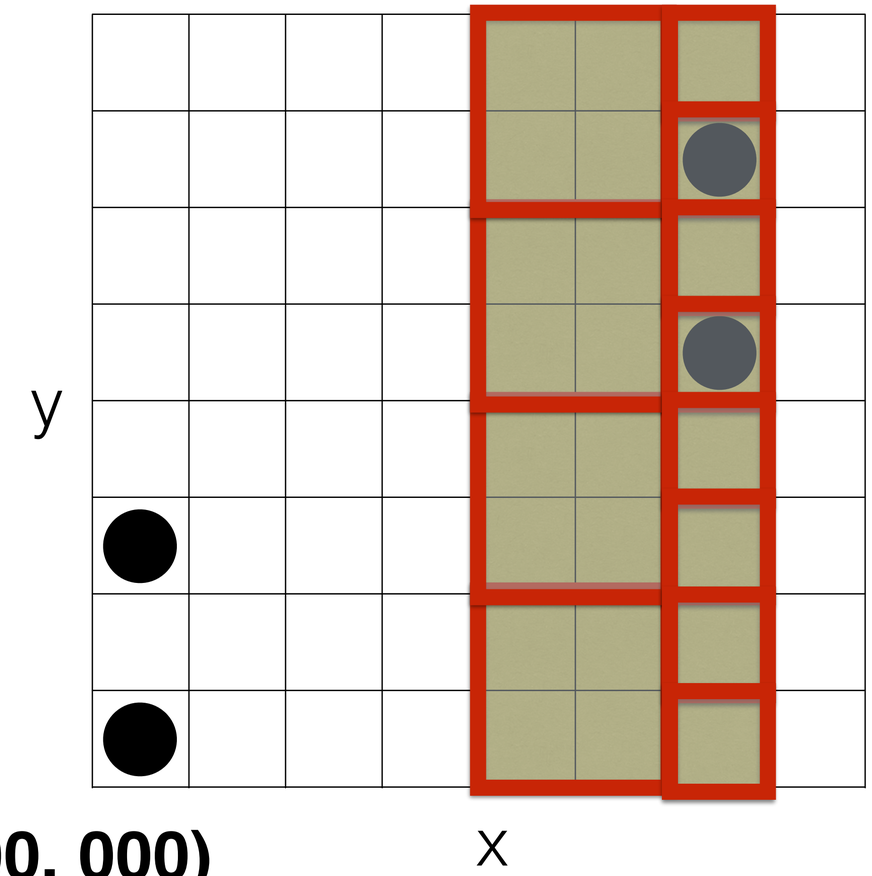
Example 5



(000, 000)
(000, 010)
(110, 100)
(110, 110)

What if we build a single dimension for both X and Y, like a quad-tree?

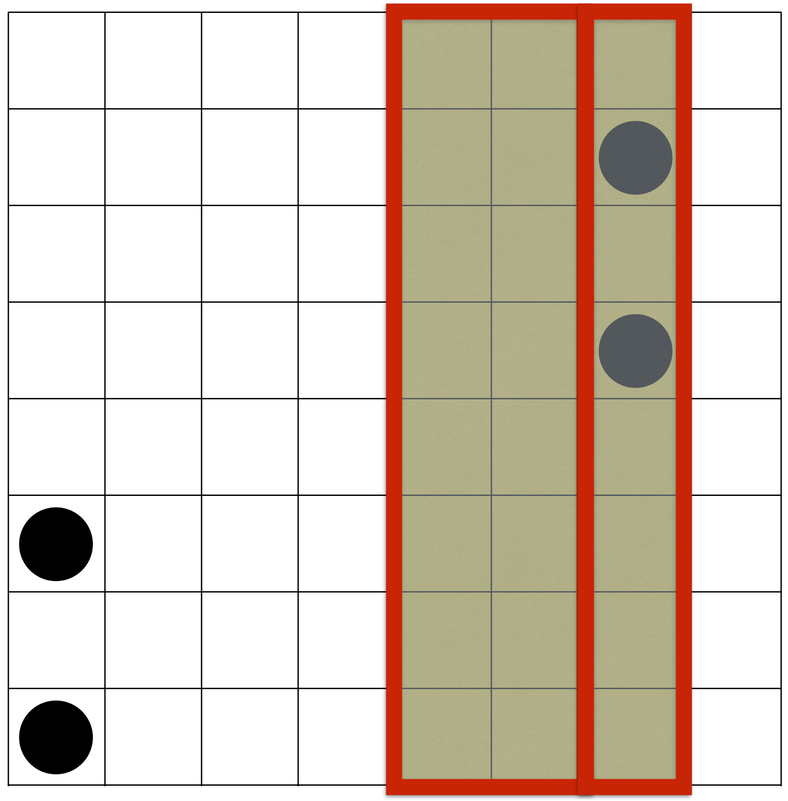
Example 5



(000, 000)
(000, 010)
(110, 100)
(110, 110)

What if we build a single dimension for both X and Y, like a quad-tree?

Example 5b



(000, 000)
(000, 010)
(110, 100)
(110, 110)