Module 8: Miscellaneous Topics

Lecture 40: Skyline Queries

Prev topic Next topic Prev page
Next page

The Lecture Contains:

Bitmap algorithm

- Assume that the domain of each dimension is discretized into n values $\{0, \ldots, n-1\}$
- Represent each value as a bitmap of size n bits
- Assume that the dominating function is $<$
- Represent a value v as a bitmap whose v MSBs are set to 0 and $n - v$ LSBs set to 1

Bitmap algorithm

- Assume that the domain of each dimension is discretized into n values $\{0, \ldots, n-1\}$
- Represent each value as a bitmap of size n bits
- Assume that the dominating function is $<$
- Represent a value v as a bitmap whose v MSBs are set to 0 and $n - v$ LSBs set to 1
- For an object, extract bit slices for appropriate dimensions
- Bit slice for bit i contains the corresponding bits of *all* objects
- Bit slice A_i corresponds to the $(v_i + 1)$-th MSB for dimension i
- Bitwise AND of A_i of all dimensions produces A
- A captures all objects that are better or equal in all dimensions

Bitmap algorithm

- Assume that the domain of each dimension is discretized into n values $\{0, \ldots, n-1\}$
- Represent each value as a bitmap of size n bits
- Assume that the dominating function is $<$
- Represent a value v as a bitmap whose v MSBs are set to 0 and $n - v$ LSBs set to 1
- For an object, extract bit slices for appropriate dimensions
- Bit slice for bit i contains the corresponding bits of *all* objects
- Bit slice A_i corresponds to the $(v_i + 1)$-th MSB for dimension i
- Bitwise AND of A_i of all dimensions produces A
- A captures all objects that are better or equal in *all* dimensions
- Bit slice B_i corresponds to the v_i -th MSB for dimension
- Bitwise OR of B_i of all dimensions produces B
- B captures all objects that are strictly better in at least one dimension

Bitmap algorithm

- Assume that the domain of each dimension is discretized into n values $\{0, \ldots, n-1\}$
- Represent each value as a bitmap of size n bits
- Assume that the dominating function is $<$
- Represent a value v as a bitmap whose v MSBs are set to 0 and $n - v$ LSBs set to 1
- For an object, extract bit slices for appropriate dimensions
- Bit slice for bit i contains the corresponding bits of *all* objects
- Bit slice A_i corresponds to the $(v_i + 1)$-th MSB for dimension i
- Bitwise AND of A_i of all dimensions produces A
- A captures all objects that are better or equal in *all* dimensions
- Bit slice B_i corresponds to the v_i-th MSB for dimension
- Bitwise OR of B_i of all dimensions produces B
- B captures all objects that are strictly better in *at least one* dimension
- Bitwise AND of A and B produces dominator list

Bitmap algorithm

- Assume that the domain of each dimension is discretized into n values $\{0, \ldots, n-1\}$
- Represent each value as a bitmap of size n bits
- Assume that the dominating function is $<$
- Represent a value v as a bitmap whose v MSBs are set to 0 and $n - v$ LSBs set to 1
- For an object, extract bit slices for appropriate dimensions
- Bit slice for bit i contains the corresponding bits of *all* objects
- Bit slice A_i corresponds to the $(v_i + 1)$-th MSB for dimension i
- Bitwise AND of A_i of all dimensions produces A
- A captures all objects that are better or equal in *all* dimensions
- Bit slice B_i corresponds to the v_i-th MSB for dimension
- Bitwise OR of B_i of all dimensions produces B
- B captures all objects that are strictly better in *at least one* dimension
- Bitwise AND of A and B produces dominator list
- Produces approximate answers when domain is quantized

Example: representation

Object	Value	Dimension 1	Dimension 2
O_1	(1,6)	0111 1111	0000 0011
O_2	(2,5)	0011 1111	0000 0111
O_3	(4,5)	0000 1111	0000 0111
O_4	(4,4)	0000 1111	0000 1111
O_5	(3,2)	0001 1111	0011 1111
O_6	(6,4)	0000 0011	0000 1111
O_7	(6,3)	0000 0011	0001 1111
O_8	(5,1)	0000 0111	0111 1111
O_9	(7,1)	0000 0001	0111 1111

Example: dominance check

Object	Value	Dimension 1	Dimension 2
O_1	(1,6)	0111 1111	0000 0011
O_2	(2,5)	0011 1111	0000 0111
O_3	(4,5)	0000 1111	0000 0111
O_4	(4,4)	0000 1111	0000 1111
O_5	(3,2)	0001 1111	0011 1111
O_6	(6,4)	0000 0011	0000 1111
O_7	(6,3)	0000 0011	0001 1111
O_8	(5,1)	0000 0111	0111 1111
O_9	(7,1)	0000 0001	0111 1111

O_1 : Bit slice A = 1 0000 0000 $\wedge$ 1 1111 1111 = 1 0000 0000

Example: dominance check

Object	Value	Dimension 1	Dimension 2
O_1	(1,6)	0111 1111	0000 0011
O_2	(2,5)	0011 1111	0000 0111
O_3	(4,5)	0000 1111	0000 0111
O_4	(4,4)	0000 1111	0000 1111
O_5	(3,2)	0001 1111	0011 1111
O_6	(6,4)	0000 0011	0000 1111
O_7	(6,3)	0000 0011	0001 1111
O_8	(5,1)	0000 0111	0111 1111
O_9	(7,1)	0000 0001	0111 1111

O_1 : Bit slice A = 1 0000 0000 $\wedge$ 1 1111 1111 = 1 0000 0000

O_1 : Bit slice B = 0 0000 0000 $\vee$ 0 1111 1111 = 0 1111 1111

Example: dominance check

Object	Value	Dimension 1	Dimension 2
O_1	(1,6)	0111 1111	0000 0011
O_2	(2,5)	0011 1111	0000 0111
O_3	(4,5)	0000 1111	0000 0111
O_4	(4,4)	0000 1111	0000 1111
O_5	(3,2)	0001 1111	0011 1111
O_6	(6,4)	0000 0011	0000 1111
O_7	(6,3)	0000 0011	0001 1111
O_8	(5,1)	0000 0111	0111 1111
O_9	(7,1)	0000 0001	0111 1111

O_1 : Bit slice A = 1 0000 0000 $\wedge$ 1 1111 1111 = 1 0000 0000

O_1 : Bit slice B = 0 0000 0000 $\vee$ 0 1111 1111 = 0 1111 1111

O_1 : Bit slice C = $A \wedge B$ = 0 0000 0000

Therefore, O_1 is a skyline object as it is not dominated by any other object

Example: dominance check

Object	Value	Dimension 1	Dimension 2
O_1	(1,6)	0111 1111	0000 0011
O_2	(2,5)	0011 1111	0000 0111
O_3	(4,5)	0000 1111	0000 0111
O_4	(4,4)	0000 1111	0000 1111
O_5	(3,2)	0001 1111	0011 1111
O_6	(6,4)	0000 0011	0000 1111
O_7	(6,3)	0000 0011	0001 1111
O_8	(5,1)	0000 0111	0111 1111
O_9	(7,1)	0000 0001	0111 1111

O_3 : Bit slice A = 1 0000 0000 $\wedge$ 1 1111 1111 = 1 0000 0000

Example: dominance check

Object	Value	Dimension 1	Dimension 2
O_1	(1,6)	0111 1111	0000 0011
O_2	(2,5)	0011 1111	0000 0111
O_3	(4,5)	0000 1111	0000 0111
O_4	(4,4)	0000 1111	0000 1111
O_5	(3,2)	0001 1111	0011 1111
O_6	(6,4)	0000 0011	0000 1111
O_7	(6,3)	0000 0011	0001 1111
O_8	(5,1)	0000 0111	0111 1111
O_9	(7,1)	0000 0001	0111 1111

O_3: Bit slice A = 1 0000 0000 $\wedge$ 1 1111 1111 = 1 0000 0000

O_3: Bit slice B = 0 0000 0000 $\vee$ 0 1111 1111 = 0 1111 1111

Example: dominance check

Object	Value	Dimension 1	Dimension 2
O_1	(1,6)	0111 1111	0000 0011
O_2	(2,5)	0011 1111	0000 0111
O_3	(4,5)	0000 1111	0000 0111
O_4	(4,4)	0000 1111	0000 1111
O_5	(3,2)	0001 1111	0011 1111
O_6	(6,4)	0000 0011	0000 1111
O_7	(6,3)	0000 0011	0001 1111
O_8	(5,1)	0000 0111	0111 1111
O_9	(7,1)	0000 0001	0111 1111

O_3: Bit slice A = 1 0000 0000 $\wedge$ 1 1111 1111 = 1 0000 0000

O_3: Bit slice B = 0 0000 0000 $\vee$ 0 1111 1111 = 0 1111 1111

O_3: Bit slice $C = A \wedge B$ = 0 0000 0000

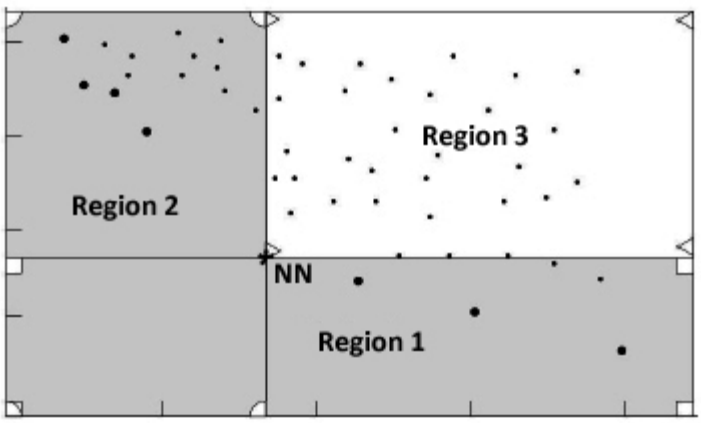
Therefore, O_3 is not a skyline object as it is dominated by O_2, O_4 and O_5

Nearest-neighbor (NN) algorithm

- Find nearest neighbor according to some monotone function
- This object is a skyline
- Exact form of monotone function does not matter

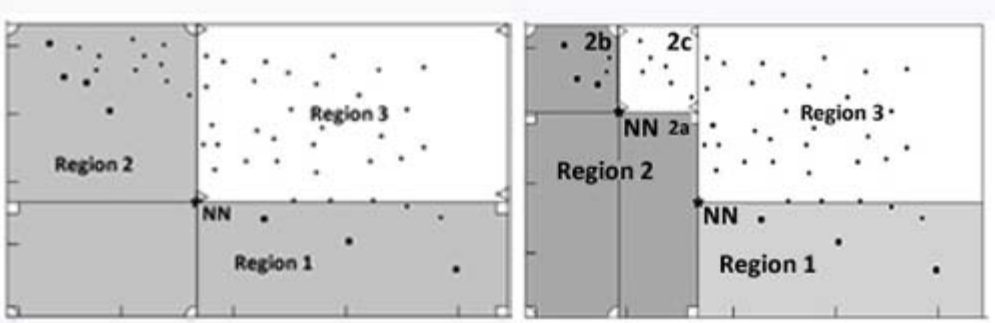
Nearest-neighbor (NN) algorithm

- Find nearest neighbor according to some monotone function
- This object is a skyline
- Exact form of monotone function does not matter
- Based on NN, split into 3 regions (for 2D)
- Region 3 can be pruned
- Recurse in regions 1 and 2



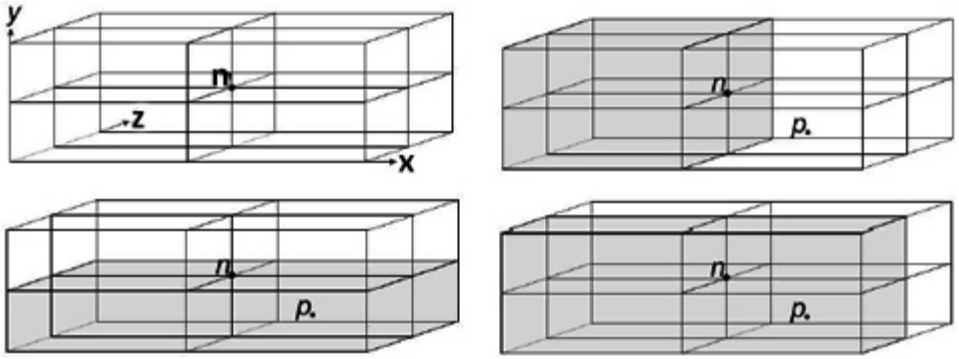
Nearest-neighbor (NN) algorithm

- Find nearest neighbor according to some monotone function
- This object is a skyline
- Exact form of monotone function does not matter
- Based on NN, split into 3 regions (for 2D)
- Region 3 can be pruned
- Recurse in regions 1 and 2



NN in higher dimensions

- In d dimensions, requires visiting d regions
- These d regions overlap; therefore, an object may be visited multiple times (object p in gure)



NN in higher dimensions

- In d dimensions, requires visiting d regions
- These d regions overlap; therefore, an object may be visited multiple times (object p in gure)
- Requires use of index structures to nd NNs