Probabilistic Aggregate Skyline Join Queries (PASJQ)

Skylines with Aggregate Operations over Existentially Uncertain Relations

Arnab Bhattacharya, Shrikant Awate

Dept. of Computer Science and Engineering,

Indian Institute of Technology, Kanpur

Motivation

- Flight from Kolkata to Los Angeles
  - No direct flight
- Through Singapore or Bangkok or Brussels or …
  - *Join* of flight relations
- Preferences
  - Low cost, low duration, better amenities, better ratings
- Aggregated preferences
  - Low *total* cost, low *total* duration but better *individual* amenities and ratings for both legs
- Flights may not take off or prices may vary
  - *Probabilistic* relations

Skylines

- User wants a better view than the full database
- If flight combination A-B is better than C-D in *all* of total cost, total duration and individual amenities and ratings, then C-D is useless
  - A-B *dominates* C-D
- If, however, A-B is better than E-F in cost but E-F is better in duration, then it is not clear
  - *Neither* A-B *nor* E-F dominates each other
- **Skylines**
  - Set of *non-dominated* objects

Problem Statement

- Find skylines from probabilistic, joined relations with preferences on both aggregate and local attributes
  - Probabilistic Aggregate Skyline Join Queries (PASJQ)

Probabilistic/Uncertain Data

- Two types of uncertainty models in databases
  - Existential
    - Tuple exists with only an *existential probability*
    - Attribute values of tuples are fixed
  - Locational
    - Tuple exists
    - Attributes assume values according to a *distribution*

fno	dur	cost	exist. prob.
11	2h 10m	162	0.8
12	2h 00m	166	0.9
13	1h 50m	173	0.5
14	1h 55m	140	0.7

fno	dur		cost	
	value	prob.	value	prob.
11	2h 10m	0.8	162	0.7
	3h 20m	0.2	215	0.3
12	2h 00m	1.0	166	0.6
			282	0.4

Skyline Probability

- In an uncertain relation, no object is a skyline for sure
  - Dominator may not exist
  - Dominating values may not exist
- Each object, thus, has a *skyline probability*
- Existential model
  - Suppose, *A* is not dominated
    - $P_{sky}(A)$ = Prob. (A exists) = e_A
  - Suppose, *B* is dominated by *C* and *D* and …
    - $P_{sky}(B)$ = Prob. (B exists) and Prob. (C does not exist and D does not exist and …) = $e_B \times ((1 - e_C) \times (1 - e_D) \times \ldots)$
  - Locational model: can be modeled as existential

Example

fno	dep	Join		Aggregate		Local		e_1
		arr	dst	duration	cost	amn	rtg	
11	06:30	08:40	C	2h 10m	162	7	5	0.8
13	12:00	13:50	C	1h 50m	173	5	3	0.5
15	09:50	10:40	C	1h 40m	220	3	2	0.8
17	17:00	20:20	C	3h 20m	183	4	4	0.9
16	16:00	17:30	D	1h 30m	230	6	6	0.3
12	07:00	09:00	E	2h 00m	166	5	7	0.9
14	08:05	10:00	E	1h 55m	140	3	5	0.7

fno	Join		arr	Aggregate		Local		e_2
	src	dep		duration	cost	amn	rtg	
21	C	09:50	12:00	2h 10m	162	6	4	0.4
23	C	16:00	18:45	2h 45m	160	4	3	0.8
26	C	16:00	18:49	2h 49m	150	2	3	0.9
22	D	17:00	19:00	2h 00m	166	4	6	0.7
25	D	16:00	17:49	1h 49m	220	3	4	0.6
24	E	20:00	21:30	1h 30m	240	5	5	0.3
27	E	20:00	21:46	1h 46m	250	3	3	0.8

Joined relation

f1.fno	f2.fno	f1.dst	f2.src	f1.arr	f2.dep	f1.amn	f2.amn	f1.rtg	f2.rtg	cost	duration	e_1	e_2	p(skyline)
11	21	C	C	08:40	09:50	7	6	5	4	324	4h 20m	0.8	0.4	0.320
11	23	C	C	08:40	16:00	7	4	5	3	322	4h 55m	0.8	0.8	0.640
11	26	C	C	08:40	16:00	7	2	5	3	312	4h 59m	0.8	0.9	0.720
13	23	C	C	13:50	16:00	5	4	3	3	333	4h 35m	0.5	0.8	0.272
13	26	C	C	13:50	16:00	5	2	3	3	323	4h 39m	0.5	0.9	0.450
15	23	C	C	10:40	16:00	3	4	2	3	380	4h 25m	0.8	0.8	0.344
15	26	C	C	10:40	16:00	3	2	2	3	370	4h 29m	0.8	0.9	0.490
12	24	E	E	09:00	20:00	5	5	7	5	406	3h 30m	0.9	0.3	0.270
12	27	E	E	09:00	20:00	5	3	7	3	416	3h 46m	0.9	0.8	0.504
14	24	E	E	10:00	20:00	3	5	5	5	380	3h 25m	0.7	0.3	0.210
14	27	E	E	10:00	20:00	3	3	5	3	390	3h 41m	0.7	0.8	0.392

Two Simple Observations

- User wants objects with skyline probability at least *p*
  - *P-skyline* query
- An object *O* cannot be part of the answer set if
  - Existence probability of *O* is less than *p*
    - $P_{sky}(O) \leq e_O < p$
  - Existence probability of a dominator *D* is greater than 1-*p*
    - $P_{sky}(O) \leq (1 - e_{D \rangle O}) < p$
- All algorithms including naïve use these two pruning rules

Skyline Probability Computation

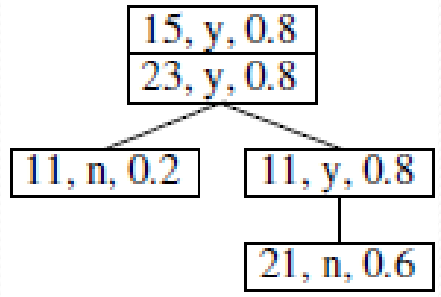
- When dominators are independent
  - $P_{sky}(O) = e_O \times \prod_{D \rightarrow O}(1 - e_D)$
- However, for flight pairs, dominators are not necessarily independent
  - Consider, 11-21
  - 11-22 may dominate
  - Also, 12-22 and 12-23 may dominate
- Tree-based mechanism

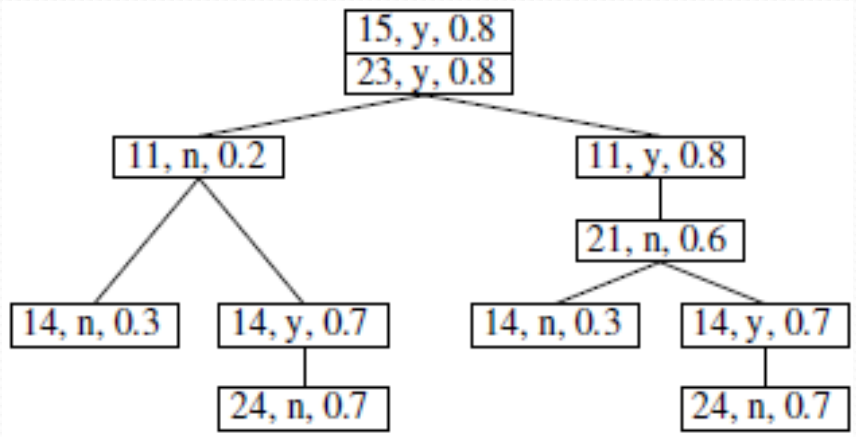
Tree

- Each node contains
  - Id of tuple
  - Flag indicating existence or absence
  - Probability
- Root contains information about joined tuple *t*
- Tuples from dominator set are added one by one
- Events along a path are independent and non-conflicting
- Non-independent events form branches
- Each path from root to a leaf encodes a possible situation in which *t* becomes a skyline
- Total skyline probability is sum along all paths

Example

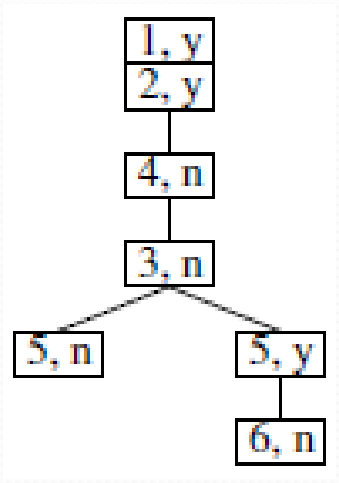
- t = (15,23) dominated by (11,21) and (14,24)

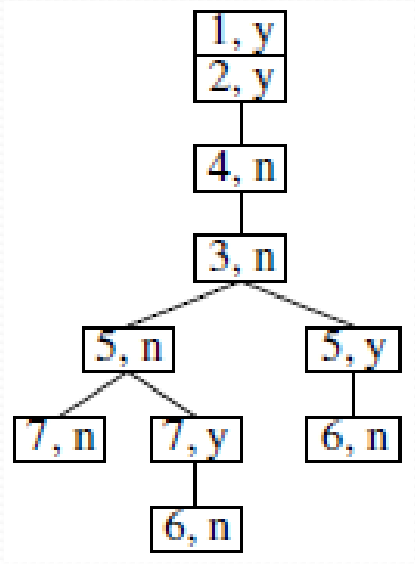


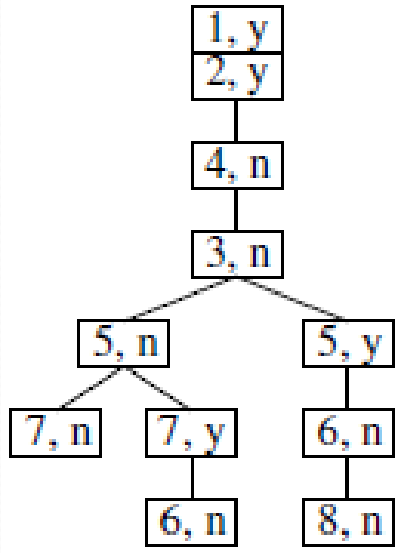


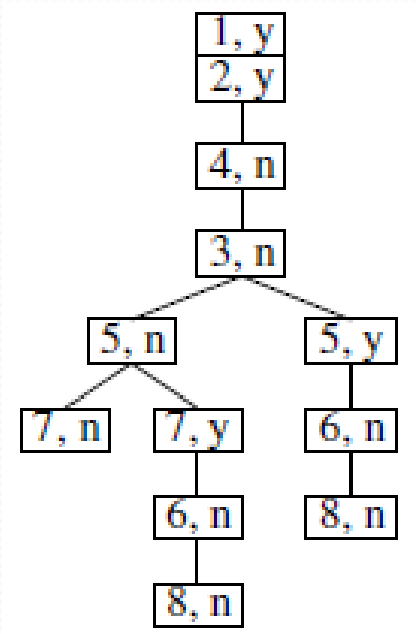
Non-independent

- (1,2) dominated by (1,4), (3,2), (3,4), (5,6), (7,6), (5,8), (7,8)









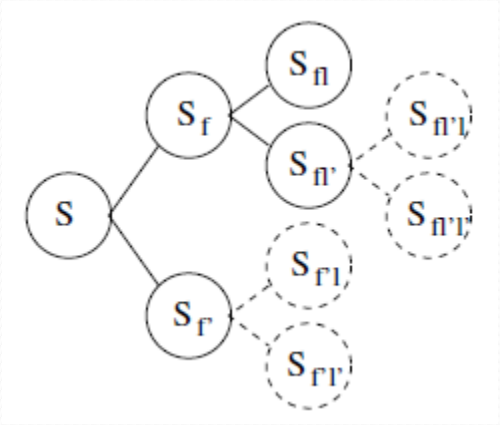
Naïve Algorithm

- Computes join of base relations
- Aggregation is part of join
- Performs P-skyline over joined relation
- Other algorithms try to push skyline operation as much as possible before join

Local and Full Dominance

- r *fully dominates* s, $r \succ_f s$, if r dominates s in *both* local and aggregate attributes
- r *locally dominates* s, $r \succ_l s$, if r dominates s in *only* local attributes
- S_f denotes *full skyline* set of S; $S_{f'}$ is its complement
- S_l denotes *local skyline* set of S; $S_{l'}$ is its complement
- S_{fl} denotes *local skyline* set of full skyline set S_f of S
- $S_{f'l'}$ denotes *non-local skyline* set of non-full skyline set $S_{f'}$ of S
- etc.

Characterization of Base Relations



Set			Flights
A_f	A_{fl}		11, 12, 16
	$A_{fl'}$	$A_{fl'l}$	13, 14
		$A_{fl'l'}$	15
$A_{f'}$			17

Set			Flights
B_f	B_{fl}		21, 22, 24
	$B_{fl'}$	$B_{fl'l}$	23, 25
		$B_{fl'l'}$	26
$B_{f'}$			27

Full Skylines

- t in $A_{fl} \times B_{fl}$ *cannot be dominated* by any joined tuple
  - Consider u in A_{fl}
  - No v in A_f dominates it locally
  - Suppose, v' in $A_{f'}$
  - Then, v in $A_f \succ_l v'$ and by transitivity $\succ_l u$
    - Contradiction
  - Thus, no u' in joined tuple t' can dominate u locally
  - Hence, no t' can dominate t either
- Through similar arguments, t in $A_{fl'} \times B_{fl}$ or $A_{fl} \times B_{fl'}$ *cannot be dominated*
- Thus, their skyline probability is *existence probability*

Target Set

- Target set for a tuple u in *base* relation
  - Set of tuples including itself *that can join and dominate* a tuple $t = u \times v$ formed using u as a component
- Target set of *joined* tuple
  - *Join* of target sets
- It reduces computation of skyline probability since dominators come from only target set

MSC Algorithm

- Uses the previous result on A_{fl}, B_{fl}, etc.

MSC		
Tuple	Target	Theorem
$u \in S_{fl}$	$\{u' \mid u'_l = u_l\}$	2
$u \notin S_{fl}$	S	-

Iterative Algorithm

- Breaks S_{fl} and $S_{f'}$ further

Iterative		
Tuple	Target	Theorem
$u \in S_{fl}$	$\{u' \mid u'_l = u_l\}$	2
$u \in S_{fl'l}$	$\{u' \mid u'_l = u_l\} \cup S_{fl} \cup S_{f'}$	3
$u \in S_{fl'l'}$	S	-
$u \in S_{f'l}$	$\{u' \mid u'_l = u_l\} \cup S_f$	3
$u \in S_{f'l'}$	S	-

Dominator-based Algorithm

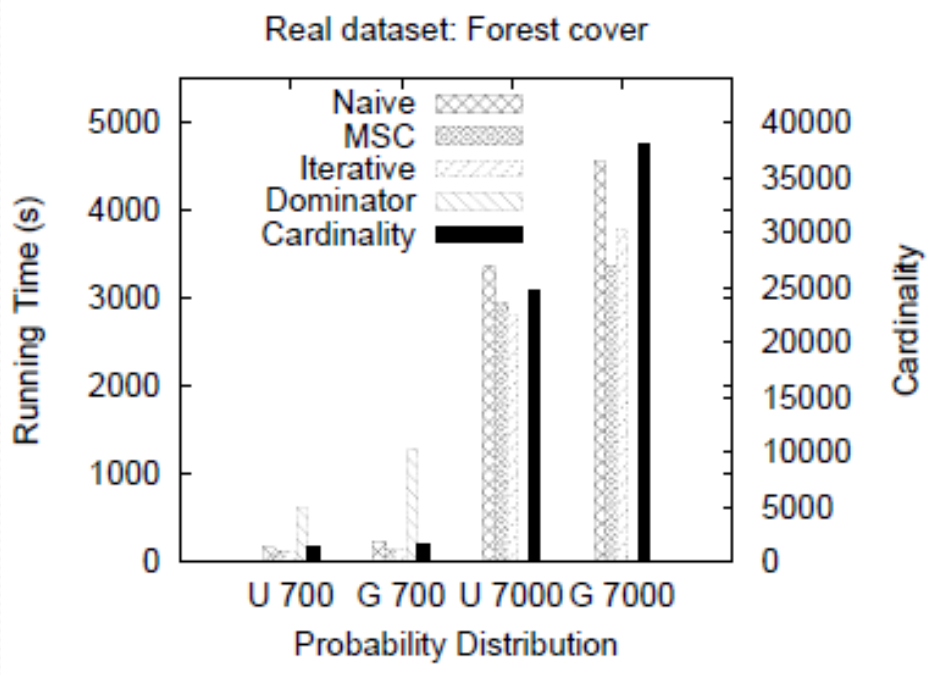
- Maintains local dominators for each tuple
- Helps reducing the target set from S

Dominator-based		
Tuple	Target	Theorem
$u \in S_{fl}$	$\{u' \mid u'_l = u_l\}$	2
$u \notin S_{fl}$	$ld(u)$	4

Experiments on Real Data

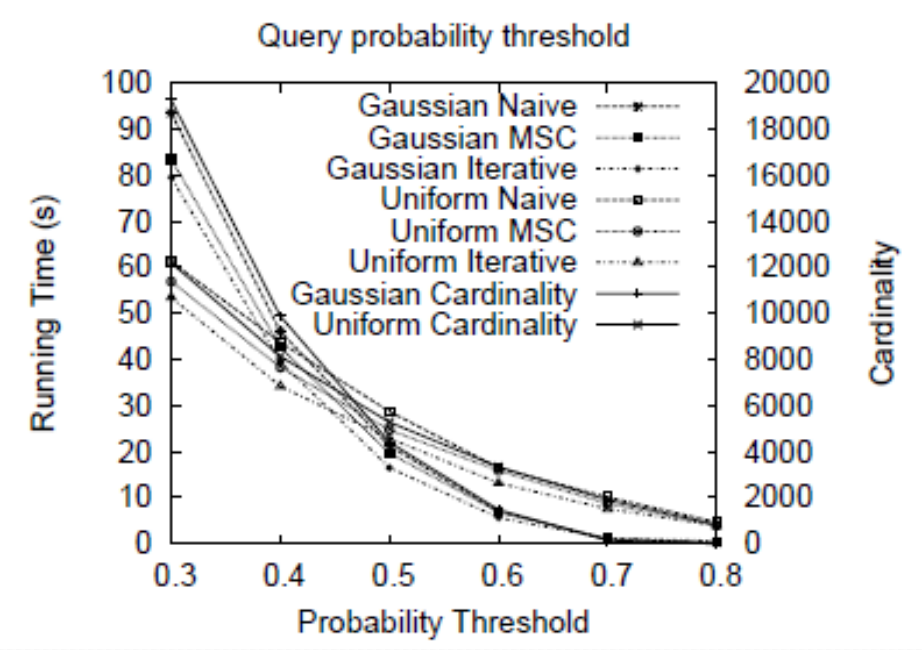
- SFS for skylines and nested-loop for join
- Covertype data
  - 3 local and 2 aggregate attributes
  - Existence probabilities simulated using Uniform U(0,1) and Gaussian N(0.6,0.15) distributions
  - Self-join with equality on covertype attribute (7)
  - Query probability threshold is 0.3
  - Two datasets
    - Full dataset of 7,000 producing 7,000,000 joined tuples
    - Reduced dataset of 700 producing 70,000 joined tuples

Results on Real Data



- Dominator-based is too impractical
- Overhead of extra computation for iterative harms it

Effect of Probability Threshold



- At high thresholds, naïve is the best
- Very few tuples survive pruning and, hence, overhead of local skyline computations is costly

Summary of Results

- Naïve is not always bad
- Gain depends critically on cardinality of full and local skyline sets
- If cardinality is low, overhead lets naïve win
- If number of attributes is more or data is anti-correlated, these sets are larger and, hence, naïve is worse
- When query threshold probability is large, very few tuples pass initial pruning and, therefore, performance is close to each other

Future Work

- Multiple relations
- Finding theoretical computational complexity of finding skylines in joined relations
- Locationally uncertain relations

THANK YOU!

Questions? & Answers!