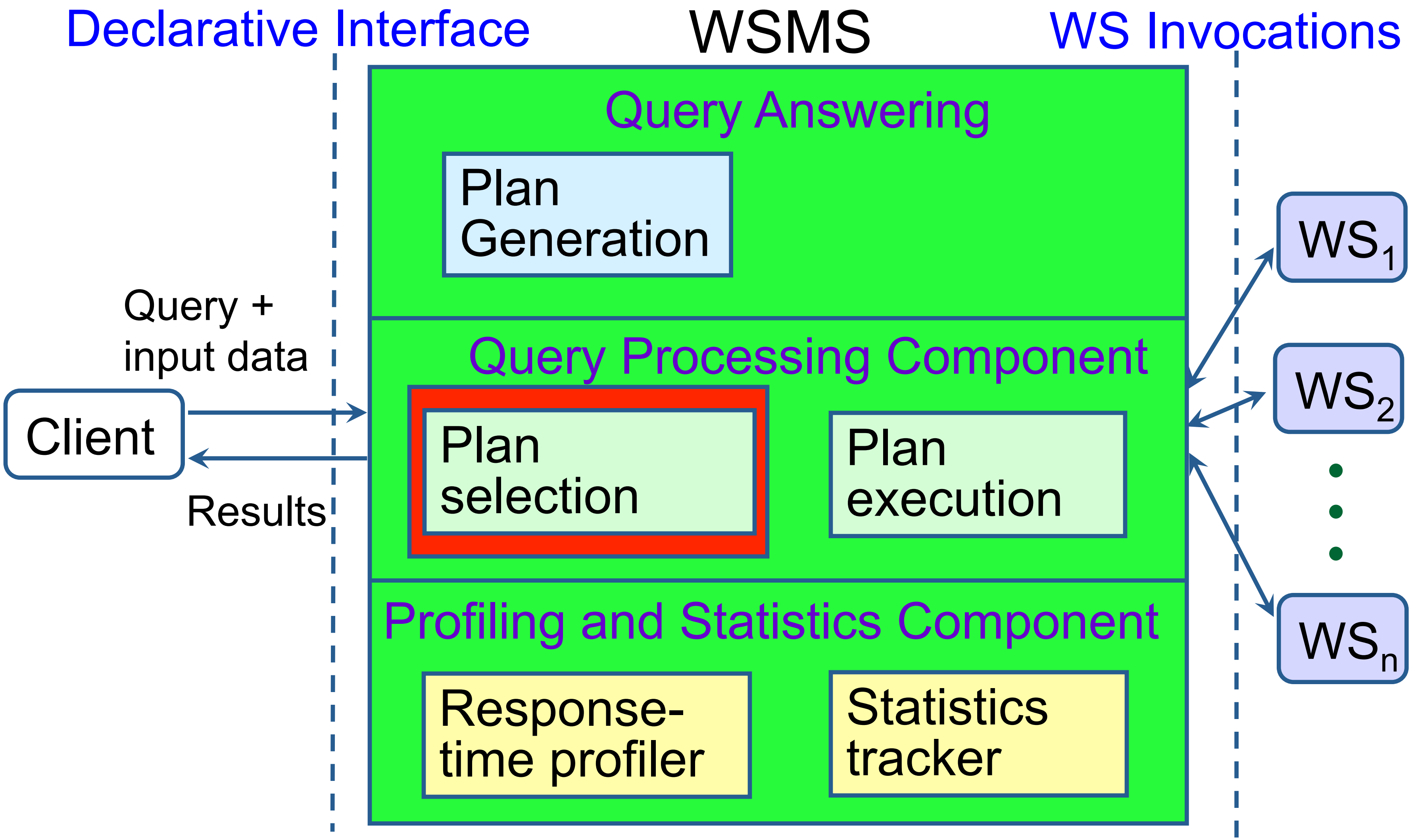
Web Services

Optimizing Web Service Compositions

WSMS Architecture



Running Example

- Credit card company wants to send offers to people with:
  - a) credit rating > 600, and
  - b) payment history = "good" on prior credit card
- Company has at its disposal:
  - L : List of potential recipients (identified by SSN)
  - WS_1 : SSN → credit rating
  - WS_2 : SSN → cc number(s)
  - WS_3 : cc number → payment history

Web Services

- Company has at its disposal:

  L : List of potential recipients (identified by SSN)

  L^{fff}(SSN, name, address)

  WS_1 : SSN → credit rating

  WS_1^{bf}(SSN, CR)

  WS_2 : SSN → cc number(s)

  WS_2^{bf}(SSN, CC)

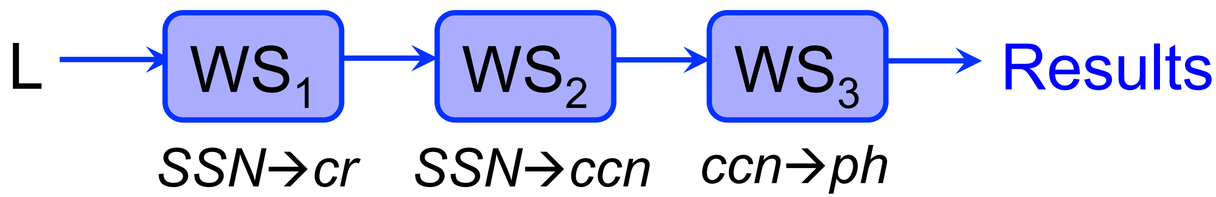
  WS_3 : cc number → payment history

  WS_3^{bf}(CC, PH)

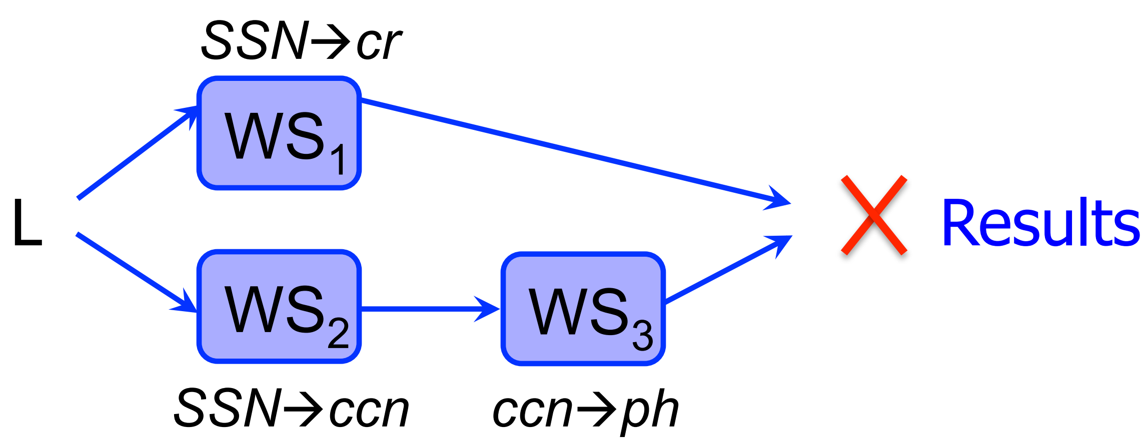
Query Evaluation

L^{fff}(SSN, name, address), WS_1^{bf}(SSN, CR), WS_2^{bf}(SSN, CC), WS_3^{bf}(CC, PH)

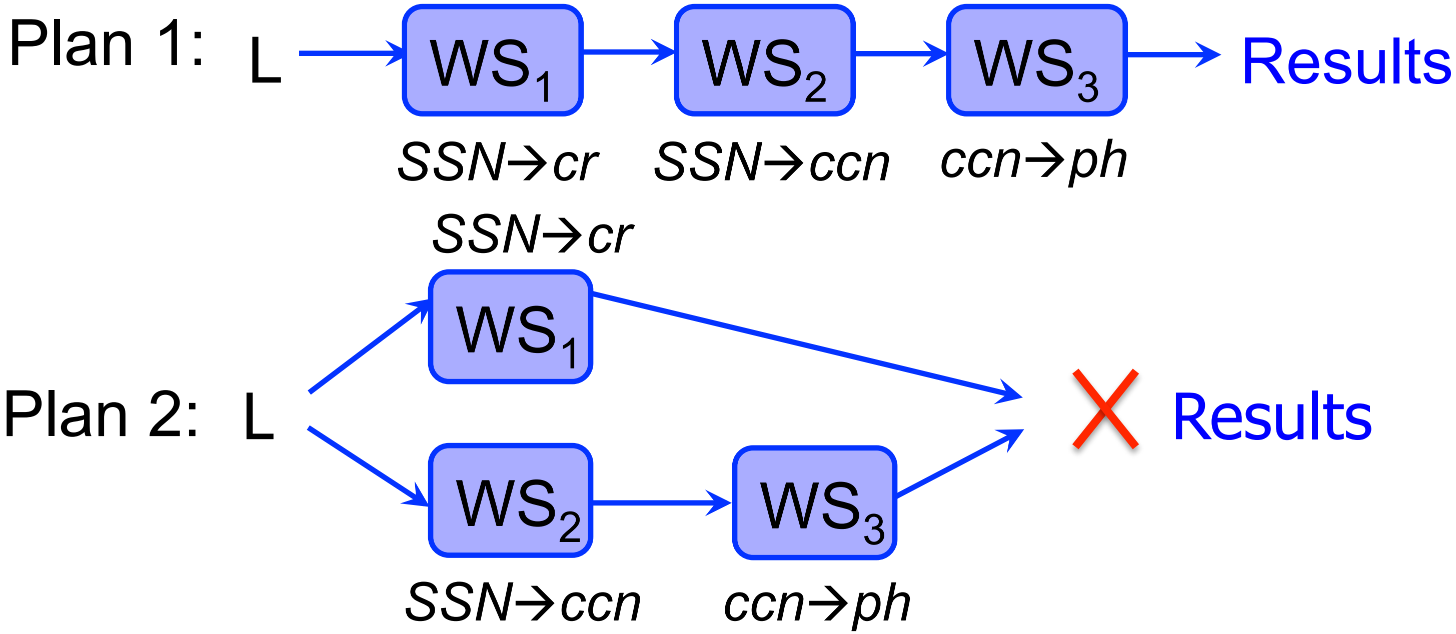
Plan 1:



Plan2:



Which plan is better?



- Assume joins are "free" (cost 0)
- Cost metric: steady-state throughput

Plan 1 is never worse

Query Optimization Primer

- Possible query plans: P_1, …, P_n
- Data/access statistics: S
- Execution cost metric: cost(P_i, S)
- GOAL: Find least-cost plan

Query Optimization Primer

- Possible query plans: P_1, ..., P_n
- Data/access statistics: S
- Execution cost metric: cost(P_i, S)
- GOAL: Find least-cost plan

Queries and Plans

Input: Evaluation plans using

- a set of extensional data sources L,
- a set of web services $WS_1, \ldots, WS_n$
- precedence constraints for WSs:

output of WS_i may be needed as input for WS_j

$$WS_i \rightarrow WS_j$$

E.g., WS_2: SSN → ccn and WS_3: ccn → ph

$$WS_2 \rightarrow WS_3$$

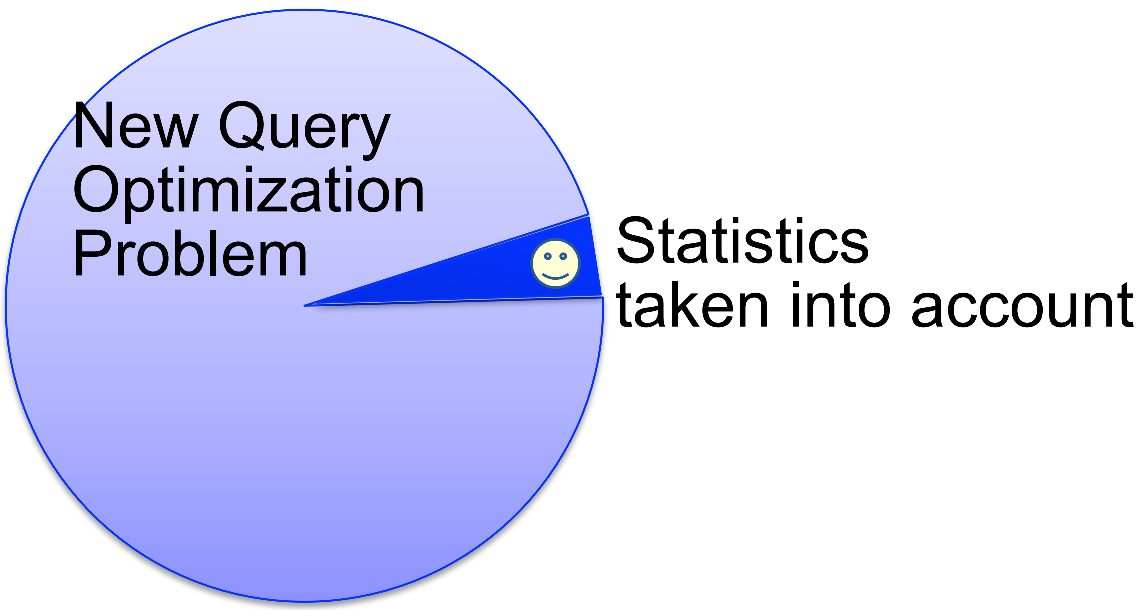
→Precedence DAG defines space of query plans

Query Optimization Primer

- Possible query plans: $P_1, \ldots, P_n$
- Data/access statistics: S
- Execution cost metric: $cost(P_i, S)$
- GOAL: Find least-cost plan

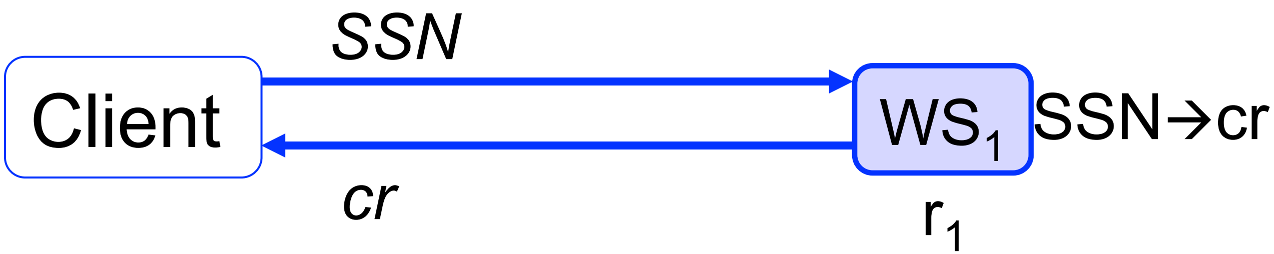
Statistics

1) WS response times
2) WS selectivity



Statistics: Response Times

r_i: per-tuple response time of WS_i from client



- $r_i \approx 1$/throughput, can be reduced by batching, parallel calls
- Assume independent response times within query plans

Statistics: Selectivity

s_i: selectivity of Ws_i

Average # output tuples per input tuple to WS_i
including post-filtering in query plan

WS_1: SSN → cr *then* filter cr > 600
If 90% of SSNs have cr > 600 then $s_1 = 0.9$

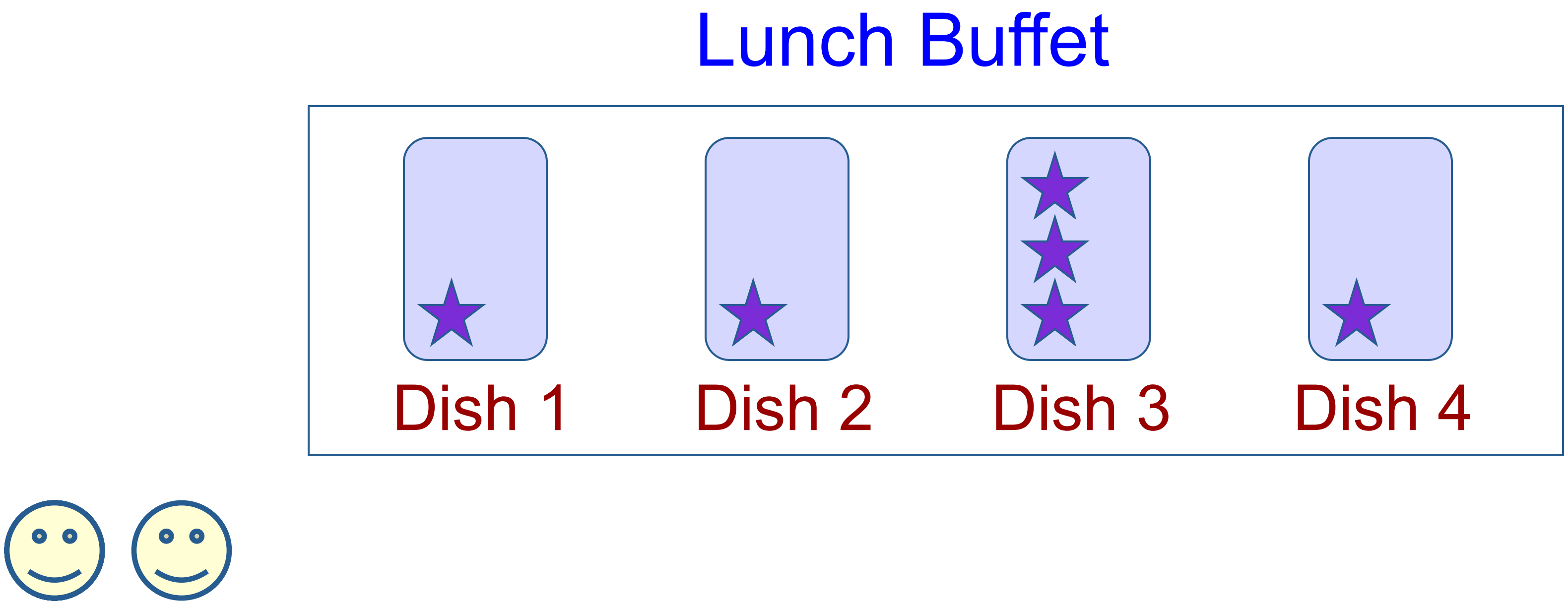
WS_2: SSN → ccn
If on average each SSN has 2 credit cards then $s_2 = 2.0$

Assumption: independent selectivities within query plans

Query Optimization Primer

- Possible query plans: $P_1, \ldots, P_n$
- Data/access statistics: S
- Execution cost metric: cost(P_i, S)
- GOAL: Find least-cost plan

Bottleneck Cost Metric



- Average *per-tuple* processing time:
  response time of the slowest (bottleneck) stage in pipeline
- Observation: assume selectivity=1 in this example

Cost Equation for Plan P

$R_i(P)$: Predecessors of WS_i in plan P

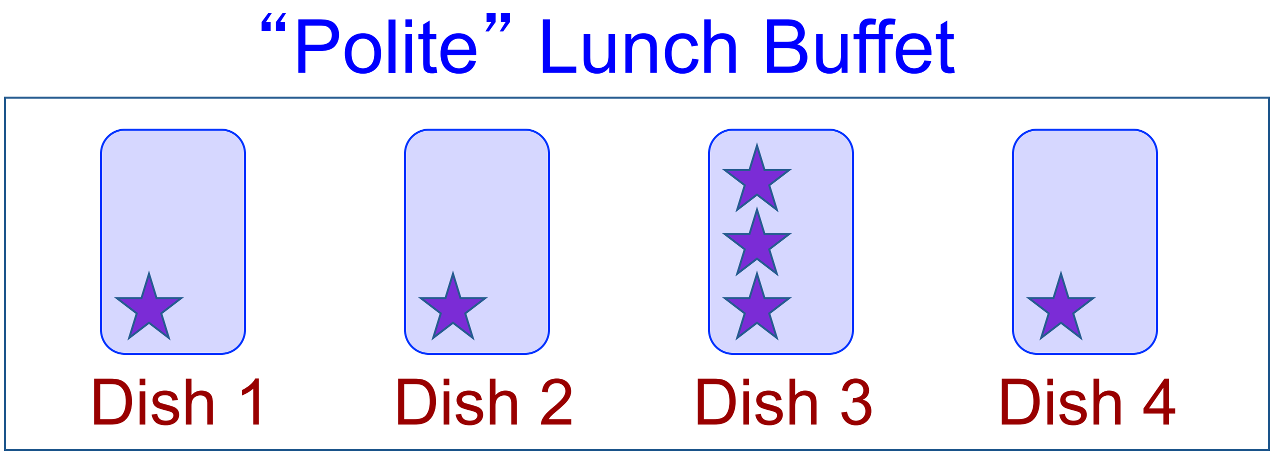
- Fraction of input tuples seen by WS_i = $\prod_{j \in R_i(P)} s_j$
- WS_i response time per input tuple = $(\prod_{j \in R_i(P)} s_j) \bullet r_i$
- Bottleneck cost metric:

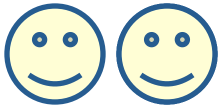
$$\text{cost}(P) = \max_{1 \le i \le n} \left(\left(\prod_{j \in R_i(P)} s_j\right) \bullet r_i \right)$$

(assumes WSMS processing is not the bottleneck)

Contrast with Sum Cost Metric

$$\text{cost}(P) = \sum_{1\leq i\leq n}\left(\left(\prod_{j\in R_i(P)} s_j\right)\cdot r_i \right)$$





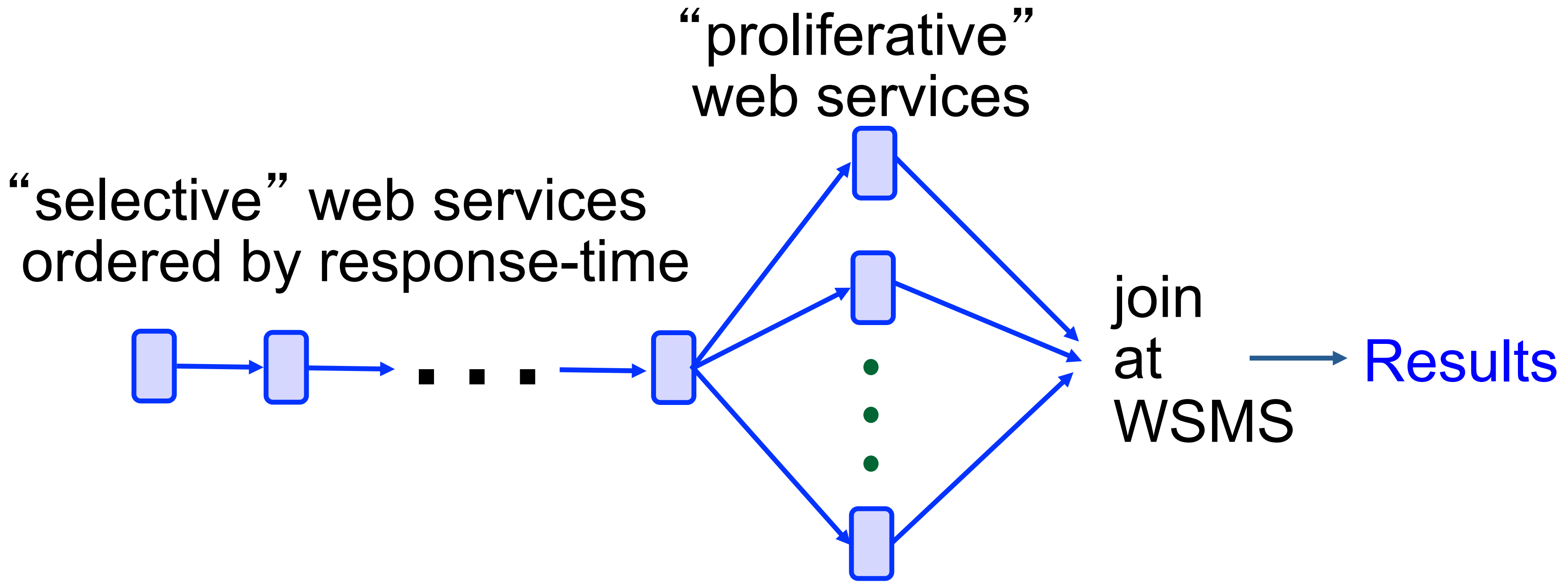
Problem Statement

- Input:
  - Web services $WS_1, \ldots, WS_n$
  - Response times $r_1, \ldots, r_n$
  - Selectivities $s_1, \ldots, s_n$
  - Precedence constraints among web services
- Output:
  - Web services arranged into a plan P
  - P respects all precedence constraints
  - cost(P) is minimized

No Precedence Constraints

Theorem: If all selectivities ≤ 1, optimal to order linearly by r_i (selectivities are irrelevant)

General case (optimal):



With Precedence Constraints

$\text{cost}(P) = \max_{1 \leq i \leq n}\left(\left(\prod_{j \in R_i(P)} s_j\right) \bullet r_i \right)$

With Precedence Constraints

$$\text{cost}(P) = \sum_{1 \leq i \leq n} \left(\left(\prod_{j \in R_i(P)} s_j \right) \cdot r_i \right)$$

Sum cost metric

- Hard to even obtain a factor $O(n^\theta)$ of optimal

With Precedence Constraints

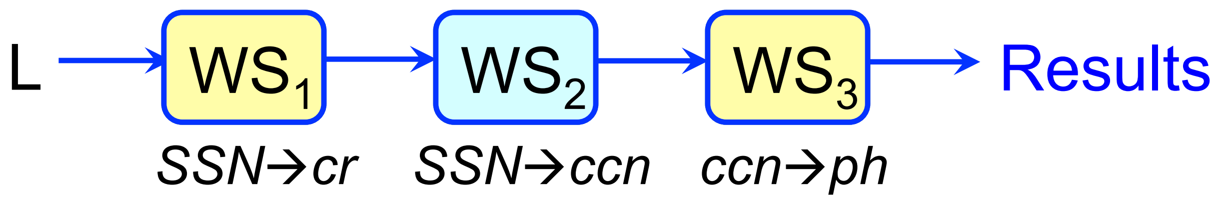
$$\text{cost}(P) = \max_{1 \le i \le n}\left(\left(\prod_{j \in R_i(P)} s_j\right) \cdot r_i \right)$$

Bottleneck (max) cost metric

- Surprisingly, optimal solution in polynomial time
- $O(n^5)$ algorithm in the paper
  - Add one WS at a time to the plan
  - WS chosen by solving a linear program

Example Revisited

Plan 1:



Plan 2:

$$\max_{1\le i\le n}\left(\left(\prod_{j\in R_i(P)} s_j\right)\cdot r_i \right)$$