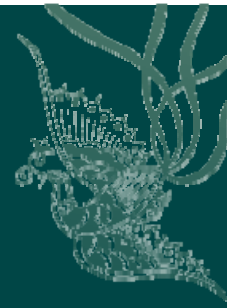
Query Prediction in Event Stream Analysis Systems

Huaiming Song

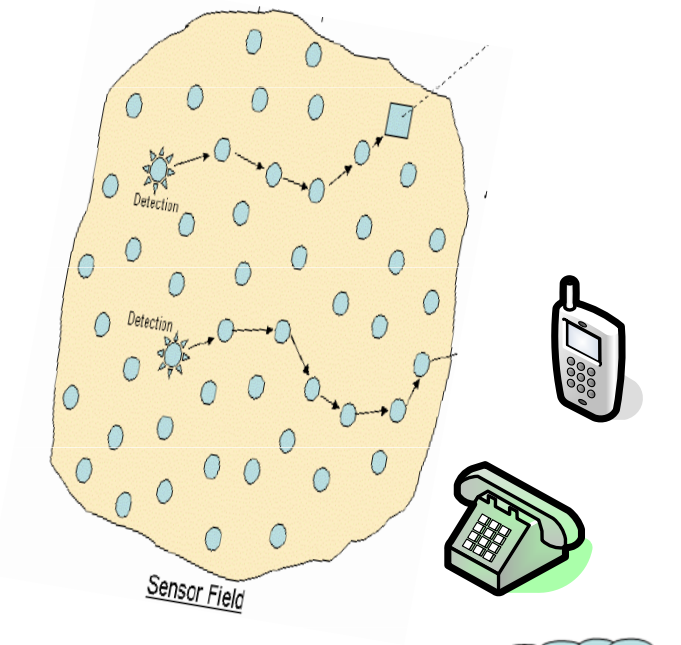
Nov.17th 2010

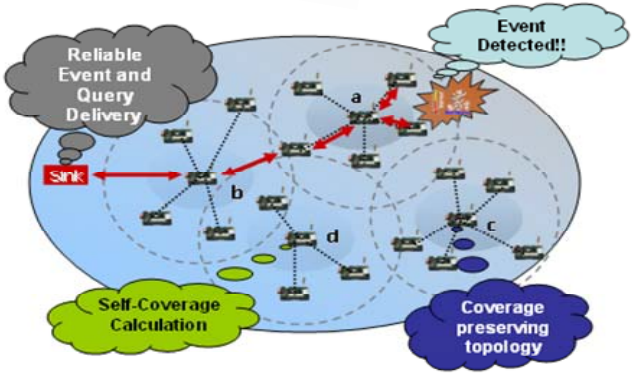
Outline



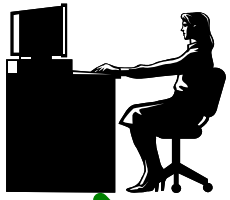
- What is Event Stream Analysis ?
- ESAS System: DBroker
- Query Prediction in ESAS

Event Stream Analysis Systems



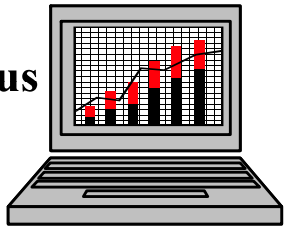


Application Systems



Register Query

Continuous Display

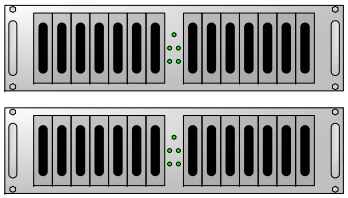


Input Stream

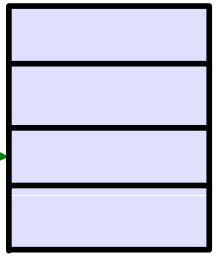
Results Stream

Event Stream

Offline Analysis

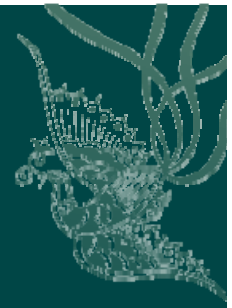


Restore Results



Event Stream Analysis System

Examples: focus on offline analysis



- Financial transaction data processing
  - Which stocks are the most active in trading market?
  - Anomaly detection in trading market…
- NetIDS
  - What is the influence of XXX BotNet over the network ?
  - What kind of systems does XXX Trojan programs attack?
- Social network mining
  - Who contact with A (maybe a criminal, or VIP)
  - How to discover a special community
    - People with same characters, and the relation ship between them?
    - How does a community evolve over time?
- Telecom monitoring system
  - What topics people talked most over the phone?
  - what is the community of a specific topic?
- Sensor network analysis system
  - …

Event Stream

- **Definition**
  - **Event: something happen or be in certain status at sometime in the observed world.** $E=<p, t>$

eg1: in Financial transaction system, an event can be represented as {t_id, t_name, stk_no, acc_no, t_prc, t_amt, t_all, serv_id, agent_id, t_time…};

eg2: in a NetIDS system, an event can be represented as {e_name,e_type, dev_id, inout_id,s_ip,s_port, d_ip,d_port,var,e_time…}

  - **Event Stream: A stream consists of endless event serials**

$$S=(\ldots E_{t-2}, E_{t-1}, E_t, E_{t+1}, E_{t+2}, \ldots)$$

  - **Event Stream Analysis Systems: Systems that make analysis or statistics of event stream.**

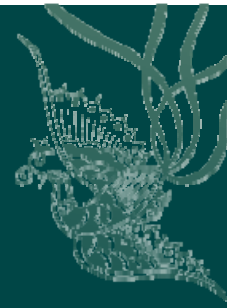
Event Stream Analyzing

- **Offline data storing and querying**
  - System characteristic：high speed arriving，append-only，no updating，detailed and statistics query，time condition，data aging
- **Two Common query types**
  - **Detailed query, Q1:**

    Select e_id, e_type, e_count, e_time from e_base where e_type =’ xxxx’ and e_time between (t1, t2);
  - **Statistics query, Q2:**

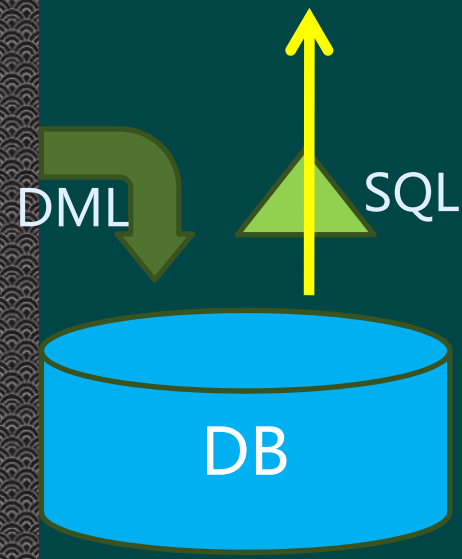
    Select e_id, sum(e_count) from e_base where e_type =’ xxxx’ and e_time between (t1, t2) group by e_id;

Features of ESAS



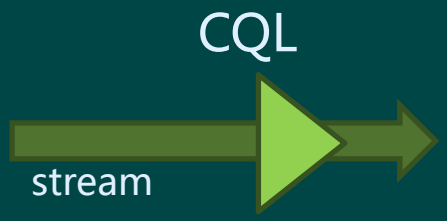
- Data Size
  - Continuous and endless stream
  - Inserting at a very high speed
    e.g.: DBroker system continuous inserting speed >20MB/s
  - Index data size even larger
- Query Processing
  - Time conditions
    - select …from…where e_time between (t1,t2)…;
  - Complex query not allowed
    - No or less join operation; no or less embed query
  - Statistics of stream is common
    - Group-by, aggregation (sum, avg, min, max, count, distinct…)

Database



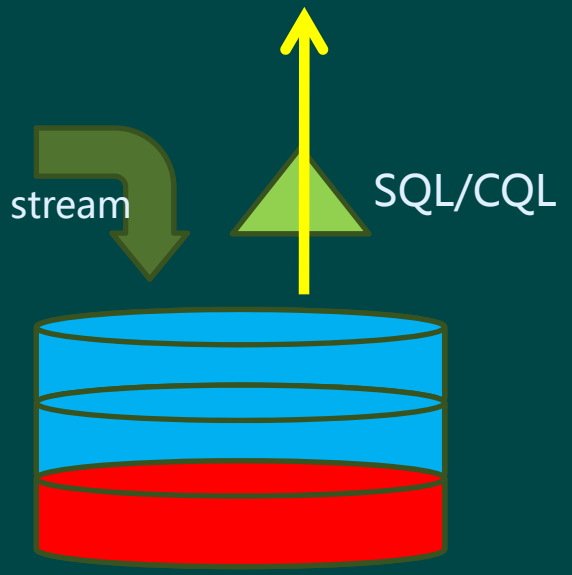
- Data write to disk
- Query from disk
- Data changes
- Ad-hoc Query

Data Stream



- Stream process
- Pre-defined query
- Continuous query
- Continuous result

Event Stream

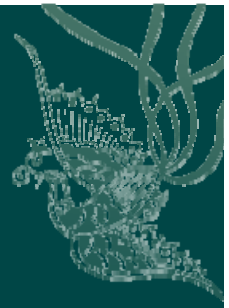


- Stream to disk
- Query from disks
- Pre-defined & ad-hoc query
- Data expired

Differences among three systems

	DBMS	DSMS	ESAS
Data model	Complex relational tables	Simple tuple serials	Simple tuple serials
Write Manner	All kinds of DML operations	Never write to Disk	Append only
Write Rate	Relatively lower	Very high	Very high
Time Character	No order, no aging problem	Time-ordered, easily fades	Time-ordered, aging problem
Query Process	Ad-Hoc query	Continuous query	Continuous and ad-hoc query
Result	Precise	Approximate	Precise and approximate

Outline

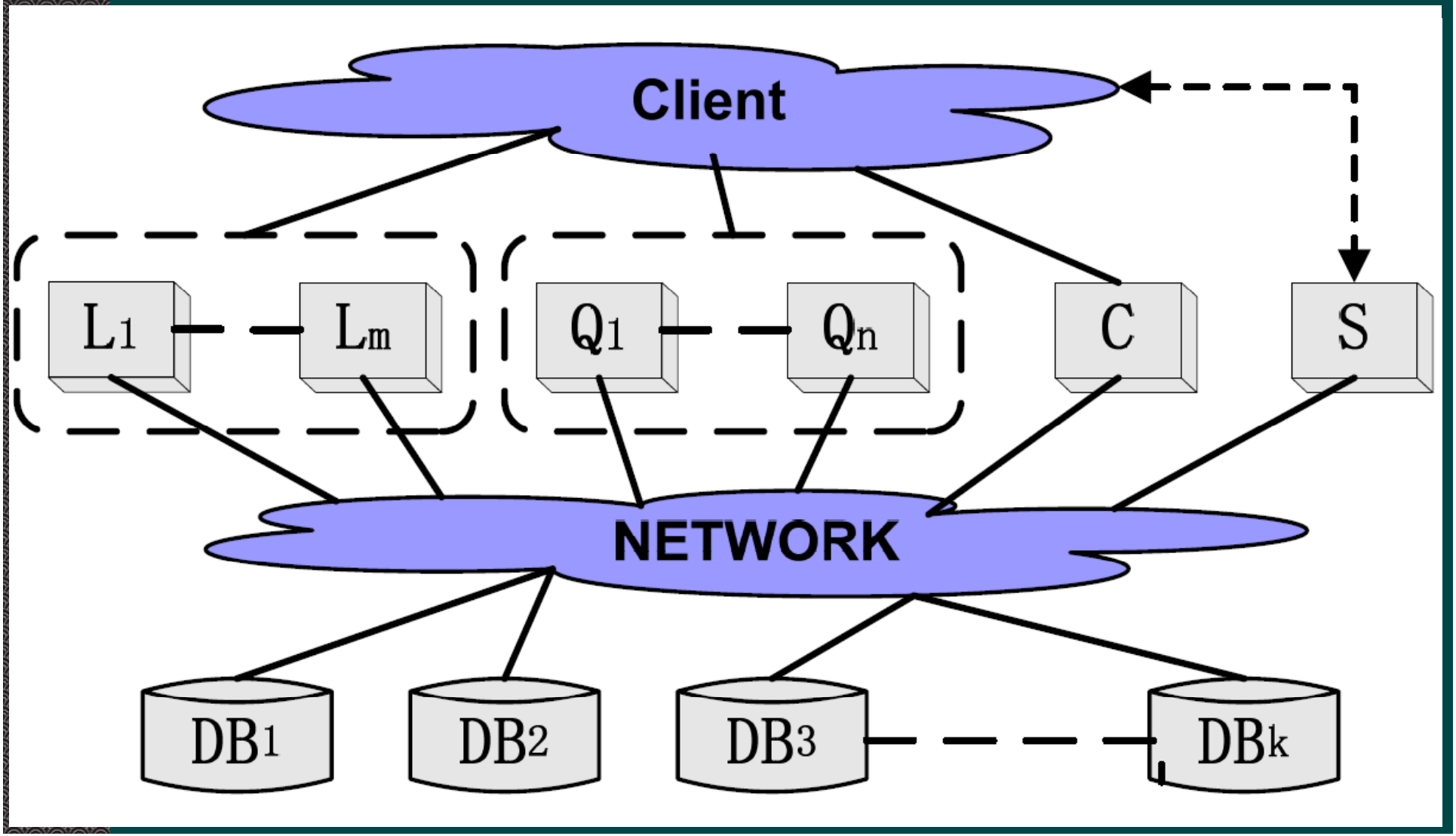


- What is Event Stream Analysis ?
- ESAS System: DBroker
- Query Prediction in ESAS

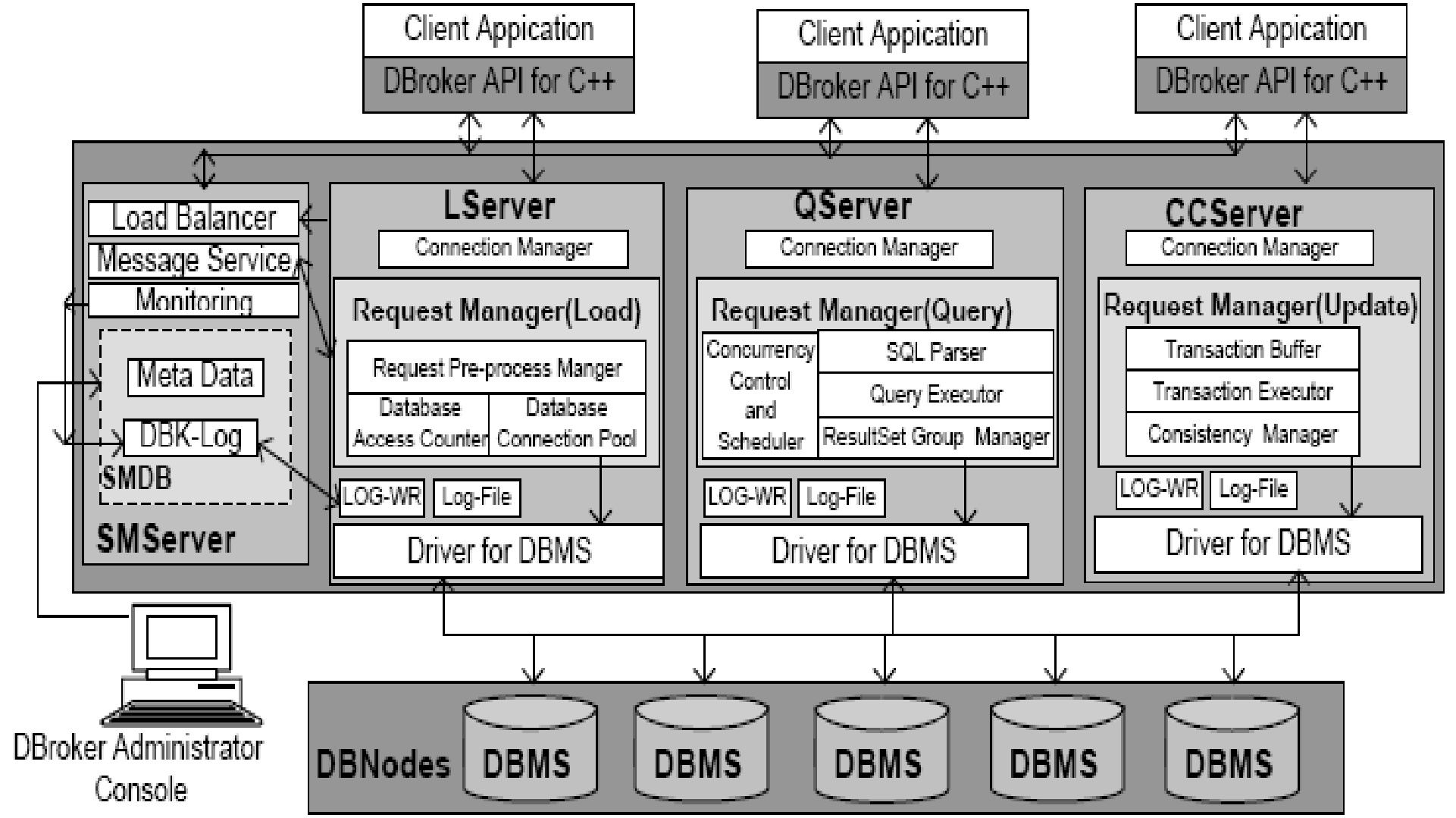
DBroker System

- Developed by ICT, CAS
- Large Scale Data Intensive Applications
  - Shared-nothing Arch.
  - Continuous Loading Speed > 20MB/s (peak loading several times larger)
  - Data size very large (hundreds TBs, or PBs)
  - Oracle 10g [RAC] in DB nodes
- ESAS applications
  - Event (e_name, e_type, dev_id, in_out_id, s_ip, s_port, d_ip, d_port, msg, e_time….)
- Event statistical or detailed query in a time window

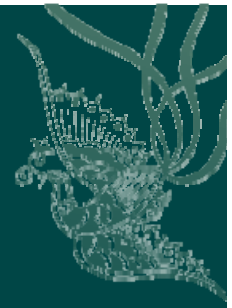
DBroker System Architecture



DBroker System Software Arch.

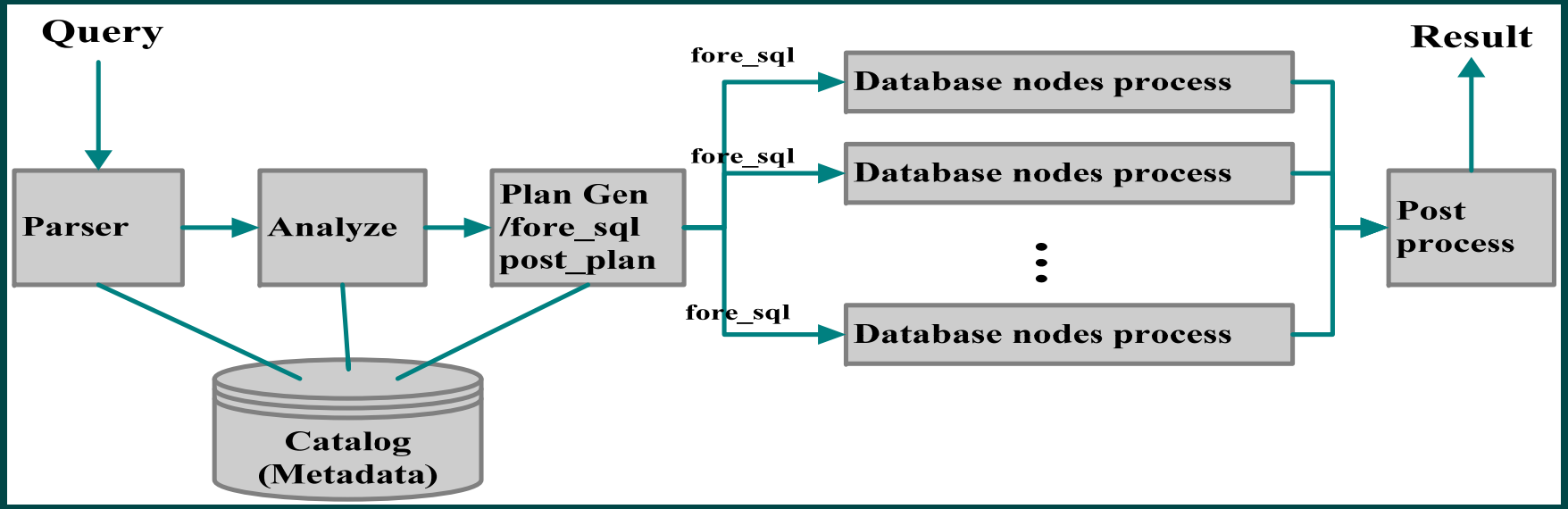


DBroker Key Technologies



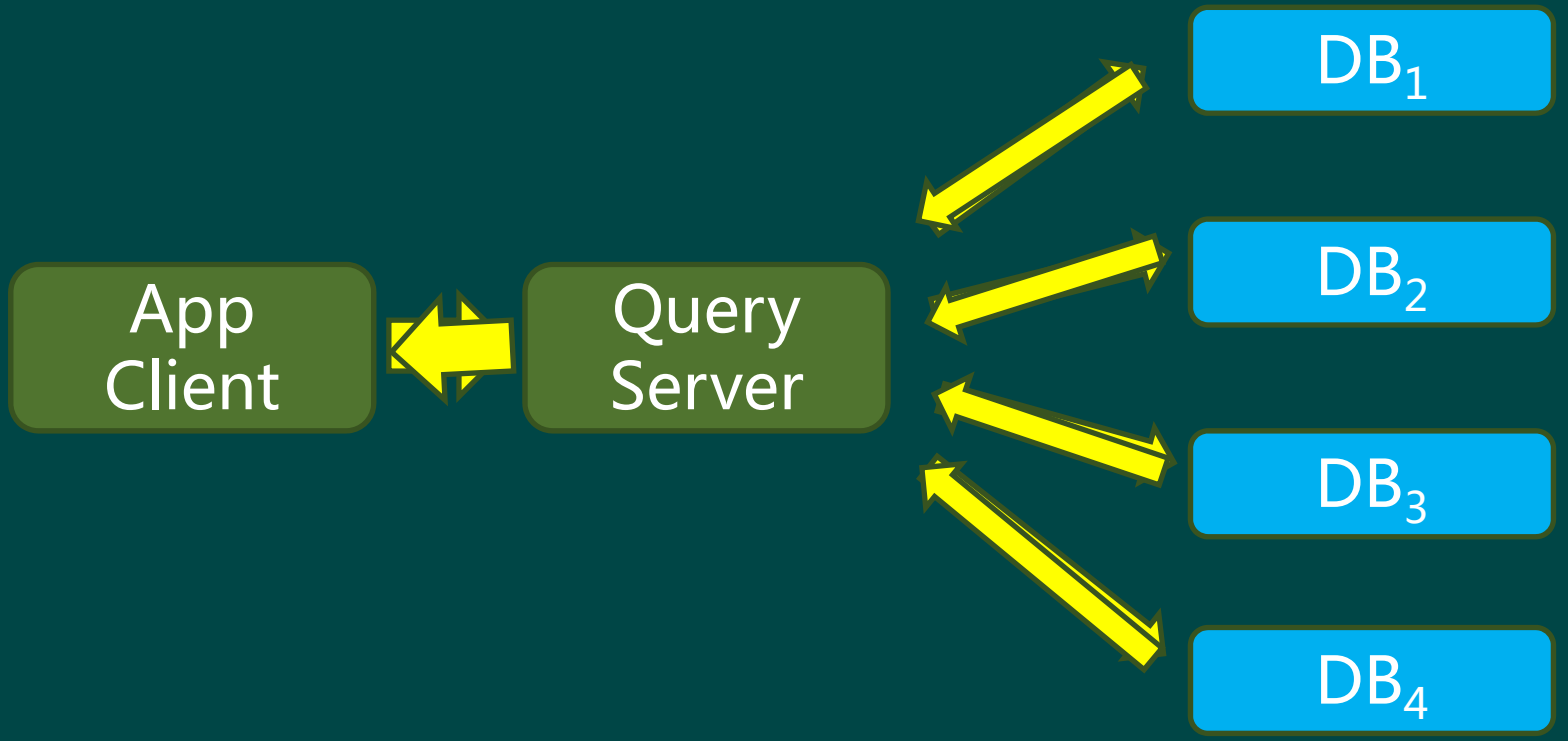
- **Functional Techs.**
  - ✓ **DB links management,2-step query process, consistence of configuration data， massage subscription**
- **Balance Techs.**
  - ✓ **Data partition, net link partition, query synchronizing, stats data repartition**
- **Performance Optimizing**
  - ✓ **Batch loading, result set pre-fetching, client buffering in transaction, lazy indexing, storage optimizing**
- **Reliability**
  - ✓ **Software: auto reconnect, auto selection of available server nodes, node fault handle of Servers**
  - ✓ **Hardware: redundancy of disks, fiber channel, network, nodes; hot spare techs.**

Query Processing



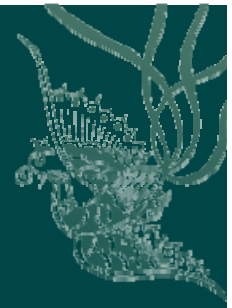
- Two-step Query Processing
  - Query distribution and processing - fore process
    - Process on all relative data nodes
    - SQL re-write
  - Result merging - post process
    - Query Server Database: global re-calculating
    - Direct manner: simple merge

Two-step Query Processing



- Detailed query: Simple merge
  - E.g.: Select * from e_base where e.type=xxx;
- Statistical query: Result re-calculation aggragate functions
  - E.g.: Select sum(e_count), e_type from e_base where … group by e_type;

Outline

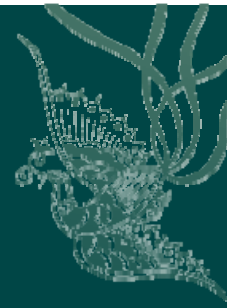


- What is Event Stream Analysis ?
- ESAS System: DBroker
- Query Prediction in ESAS
  - 3.1 Data access locality problem in ESAS
  - 3.2 Query Prediction
  - 3.4 Conclusion

Data access locality in ESAS

- A simple query
  - Select e_id, sum(e_count) from e_base where property_x= ‘valuex’ and e_time between(t1,t2) group by e_id;
- Access Locality in ESAS
  - Time condition locality
    - Most queries focused on events in a recent time window
    - [Jiao07] b-c-f(t), [Liu07] negative exponential distribution
  - Attributes values locality
    - Most queries focus on a few attributes values
    - E.g.: the most 135 frequent types of events in DBroker system (in all, about 2300 types）, their access proportion is 86.27%，78.11% of events in all requests will be queried again in three days（2006.5-2006.10

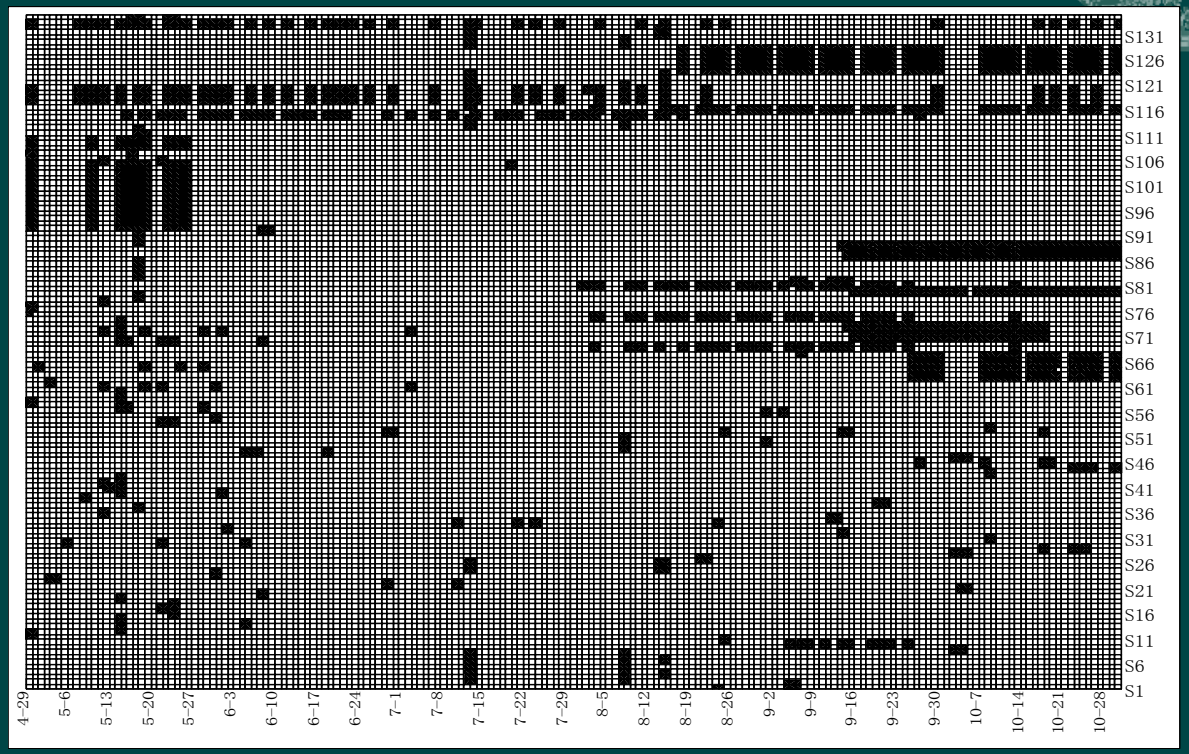
Data access locality in ESAS



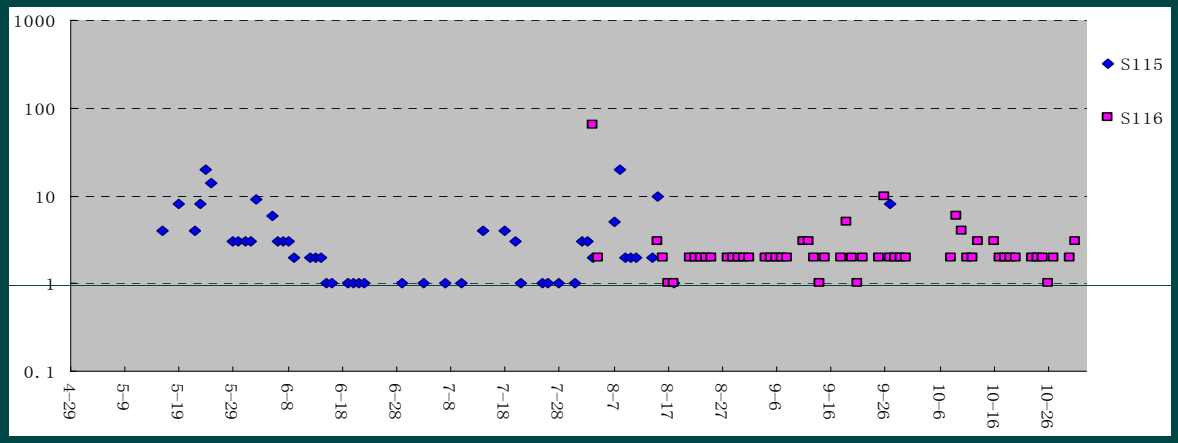
- We call access locality in ESAS: Recent hot spot event access
- Data aging problem
  - Stream flowing fast, results evolve fast
  - Same conditions, different results
- Hot spot drafting
  - Conditions evolve over time
  - Need to predict the hot spot over time

Locality of Event Stream Accessing

- DBroker system query type statistics (2006.5~2006.10)



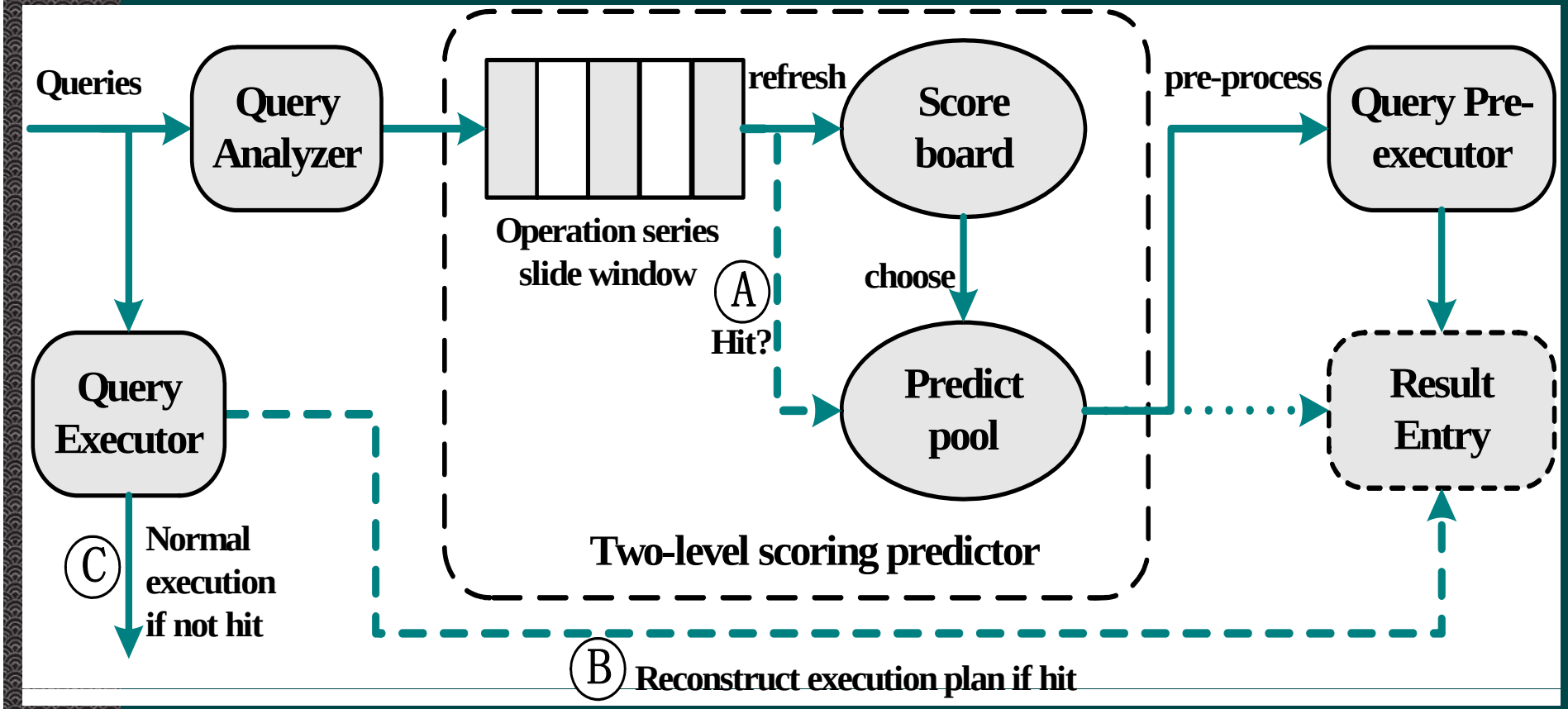
- Type 115&116 read frequency statistics



3.2 Query prediction

- Hot spot event access provide a chance for query prediction
- Common Query patterns detection
- Prediction query conditions in the future by analyzing the past conditions
  - Time window length
  - Hot Predicate
- Pre-execution the predicted query
  - Two execution strategies: global & local
  - Execution optimization

Query prediction model



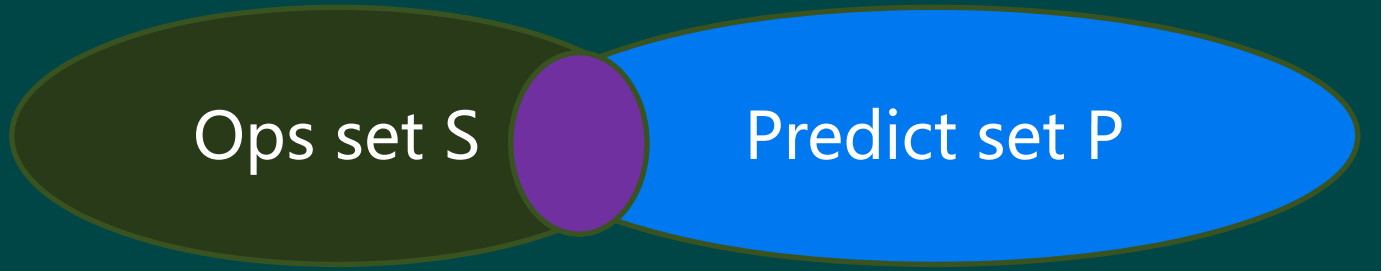
Basic Concepts

- Prediction hit
  - When a new query comes, it will be divided into several query operations. If at least one of the operations has been already in 'predict pool', then it can be called prediction hit
- Prediction invalid
  - One operation in predict pool, but not hit by any query in the one time window
- Accuracy rate (or hit rate)
  - The number of hit query / the number of all query in one time window

$$acc = |P \cap S| / |S|;$$

- Efficiency rate
  - The number of hit operation / the number of all operation in the predict pool

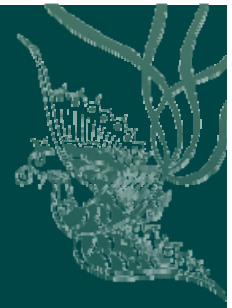
$$eff = |P \cap S| / |P|$$



Query request analyzer

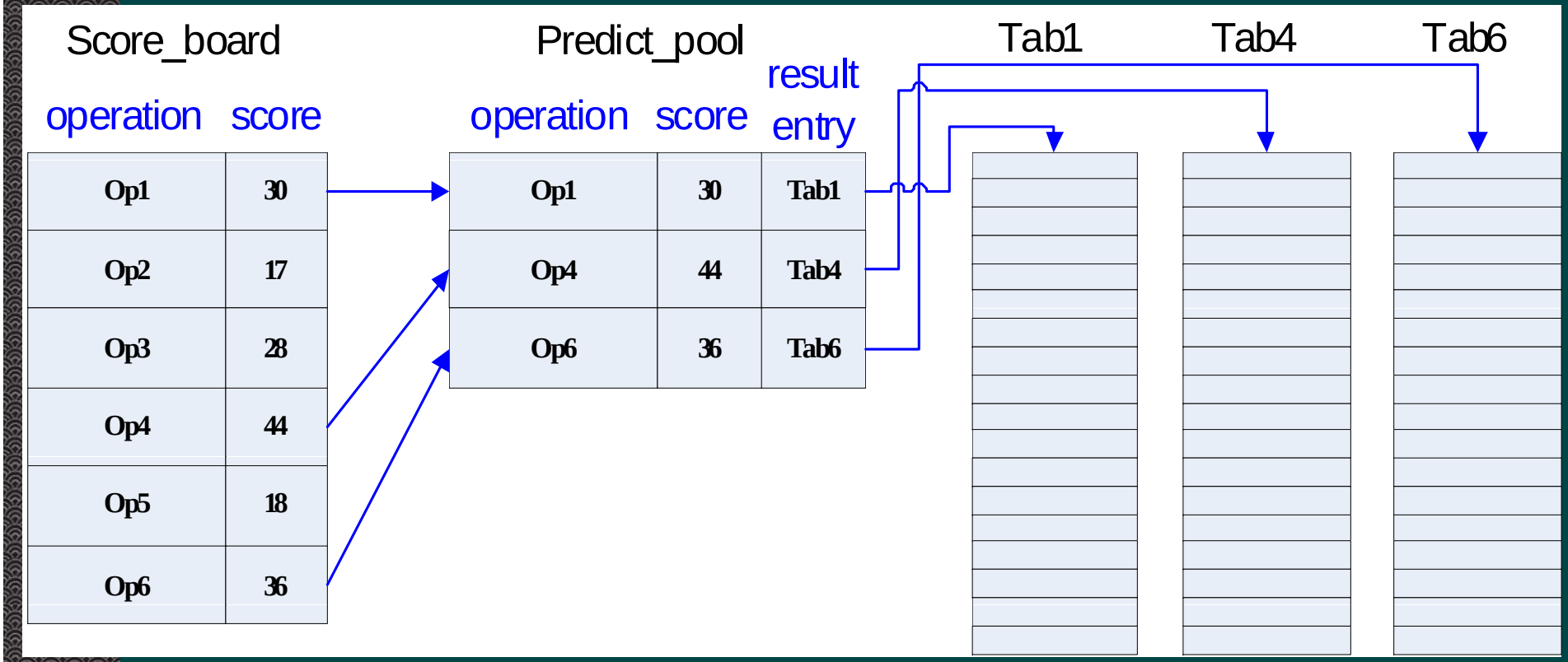
- Query operation
  - SQL divided into a set of operations
  - parser by key words
    - select, from, where, order, group…
  - Data & calculation semantics
- Time window division
  - Predict operations may arriving in next time window via analyzing historical operations in past time windows
  - Division consideration
    - Requests arriving regular pattern
    - System resource usage rate

Scoring & Prediction Unit



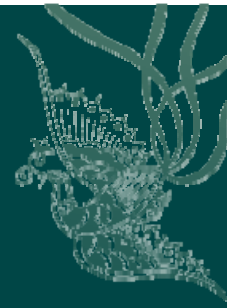
- Credit of operations
  - To measure the frequency of an operation in recent time windows, credit score
  - Higher score means higher frequency the operations appeared in recent windows
    - Regular operations，burst operations
- Score board
  - An operations collection to record all their scores
- Prediction pool
  - An operations collection with high score in score board
  - Predicted operations may come in the next time window and will be pre-executed

Score board & prediction pool



Sketch of score board, prediction pool and result entry

Two-level scoring idea



- Main idea of TLS
  - Every moment a new time window slides, modify operation credit in score_borad
    - Hit，score +hit_bonus
    - Invalid，score –invalid_penalty
  - Refresh prediction pool
- Optimizing with a decay factor
  - Every time a new time window slides, all operation credit in score_board will be multiplied by an decay factor(<1)
    - Convergence of credit score increasing
    - **init_score < bonus/(1-ε)**

Two-level scoring algorithm

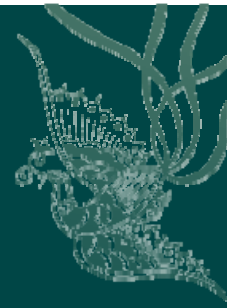


TABLE II. TWO-LEVEL SCORING ALGORITHM (TLS)

Input: op_window

Output: predict_pool

(1) For each op in score_board;

a) if op in op_window, then score_board[op].score += bonus;
b) else score_board[op].score -= penalty;
c) if score_board[op].score < score_board.threshold, then score_board.erase(op);

(2) For each op in op_window

if op not in score_board, then score_board.add[op]; /* *add new operations to score board with a initial score*/*

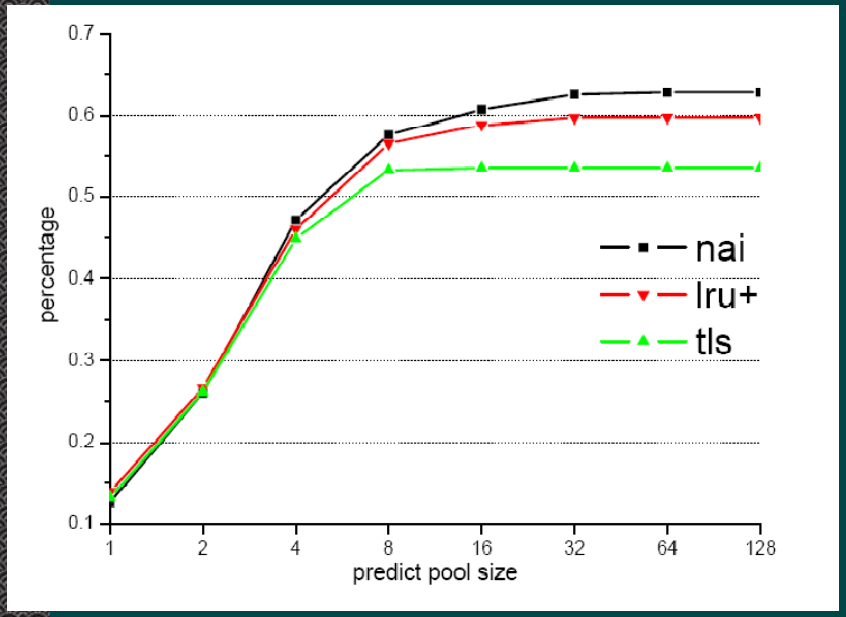
(3) predict_pool.clear(); /* *clear the prediction pool* */

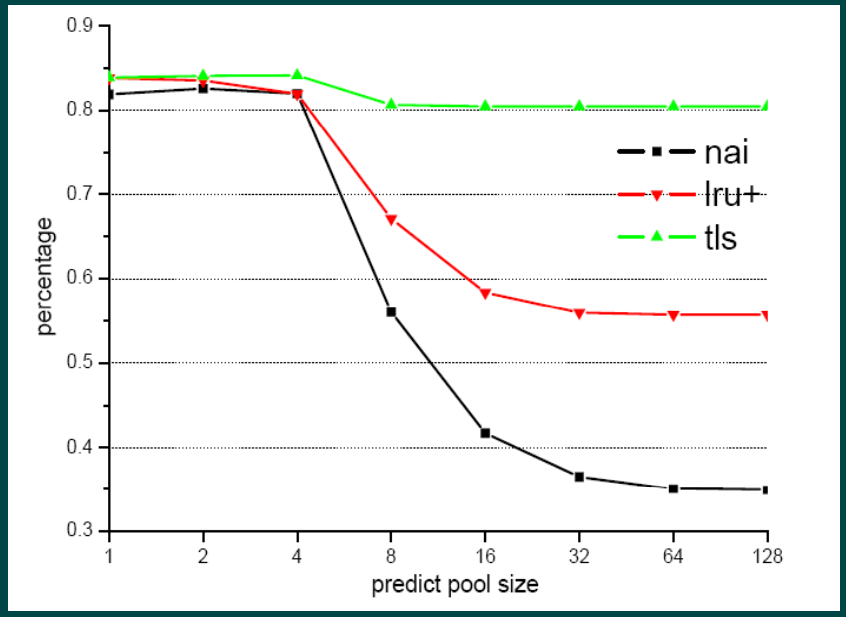
(4) Choose high scored operations for prediction pool from score_board;

d) while predict_pool.size()< MAXSIZE
e) op=score_board.getnextTop(); /* *get the operation with the rest highest score*/*
f) if op.score >= predict_pool.threshhold, then predict_pool.add(op);
g) else break;

(5) Return predict_pool;

Performance Evaluation

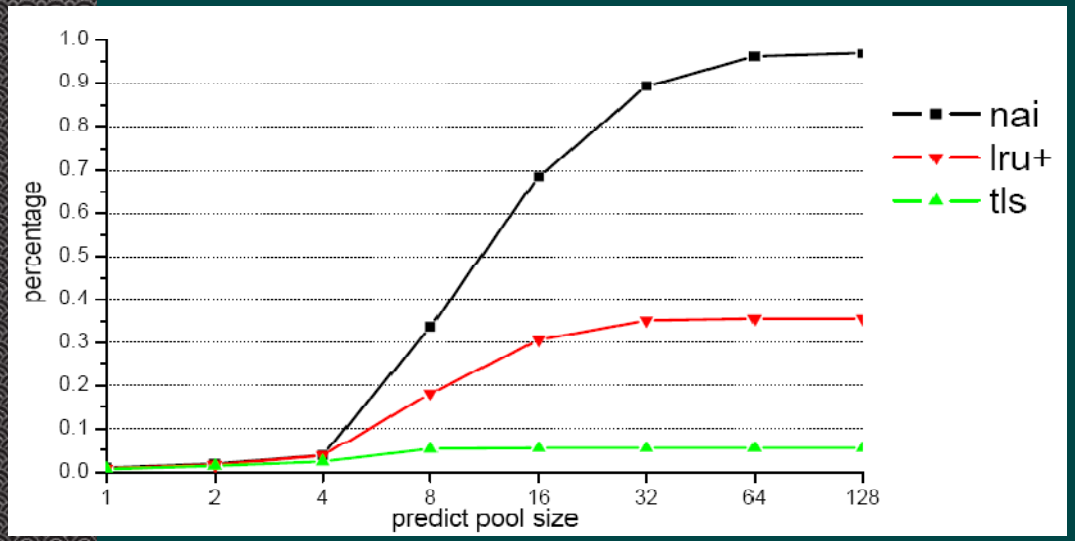




(a) prediction accuracy rate **(b) prediction efficiency rate**

Accuracy & efficiency rate of three algorithms

Performance Evaluation 2

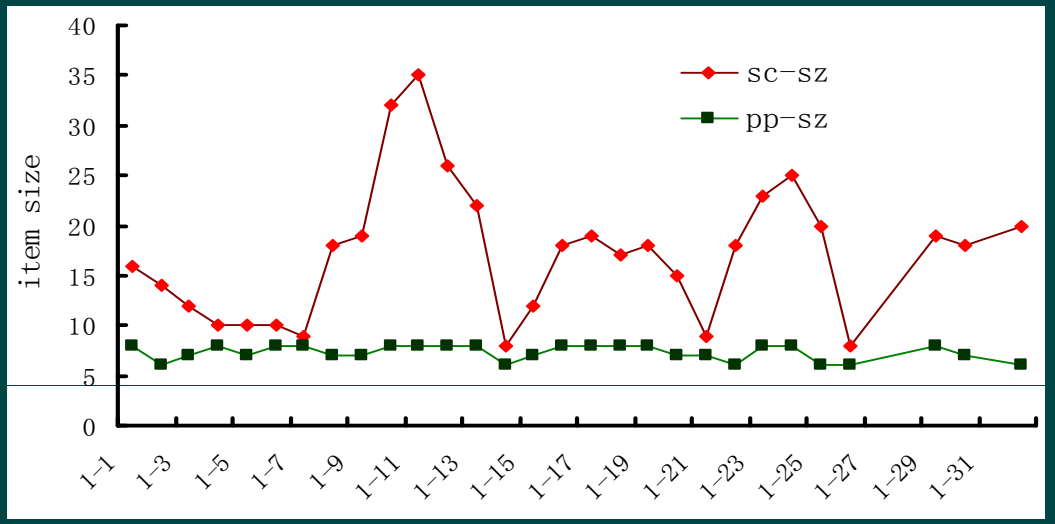


System extra workload of three algorithms。

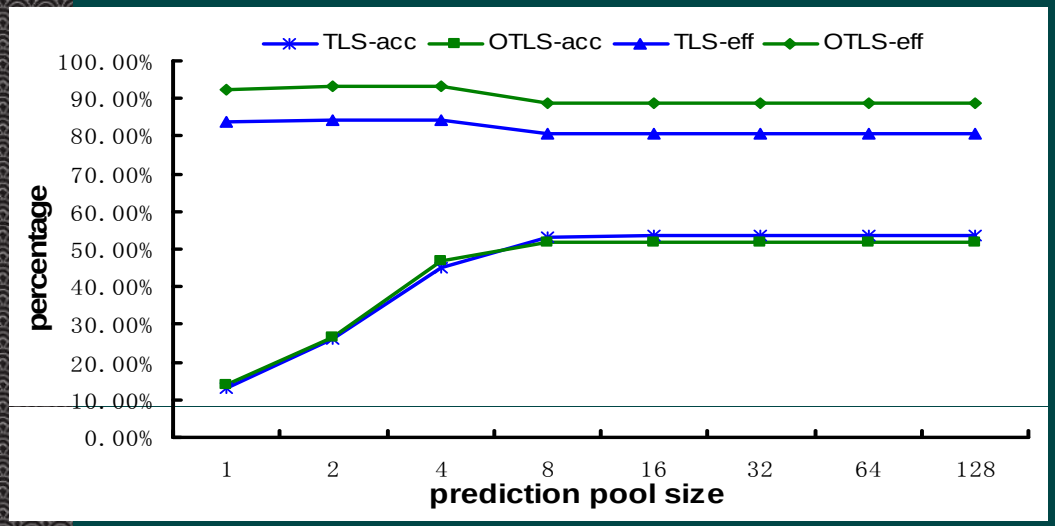
Percentage of extra requests number

Operations number in SB and PP in TLS algorithm

In which max prediction pool size is set to 8

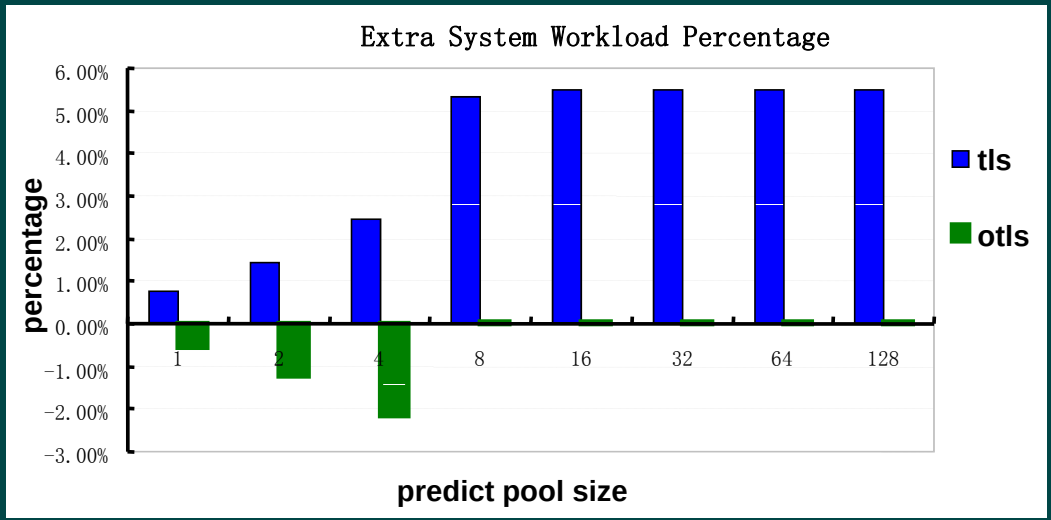


Performance Evaluation 3



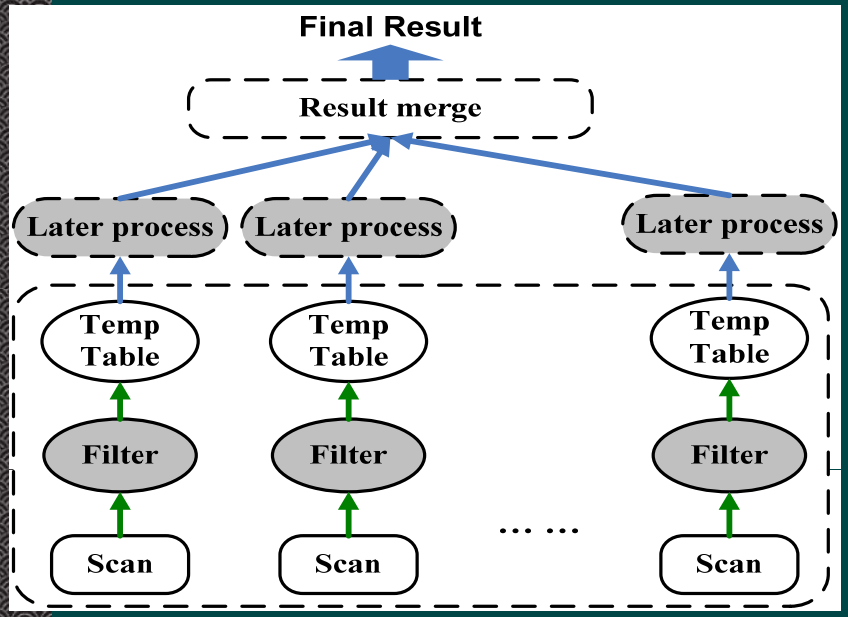
Accurate and efficiency rate while introducing decay factor in TLS

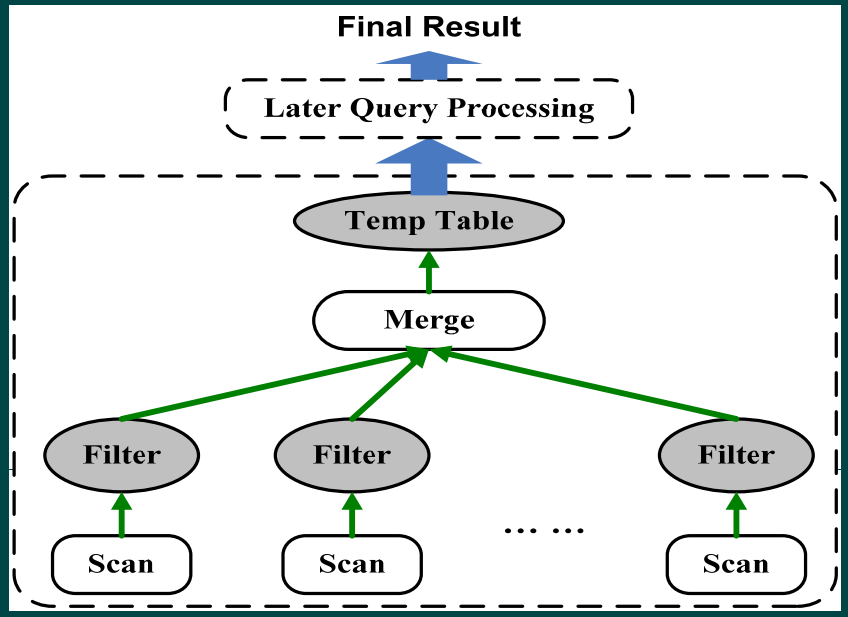
Extra system workload while introducing decay factor in TLS



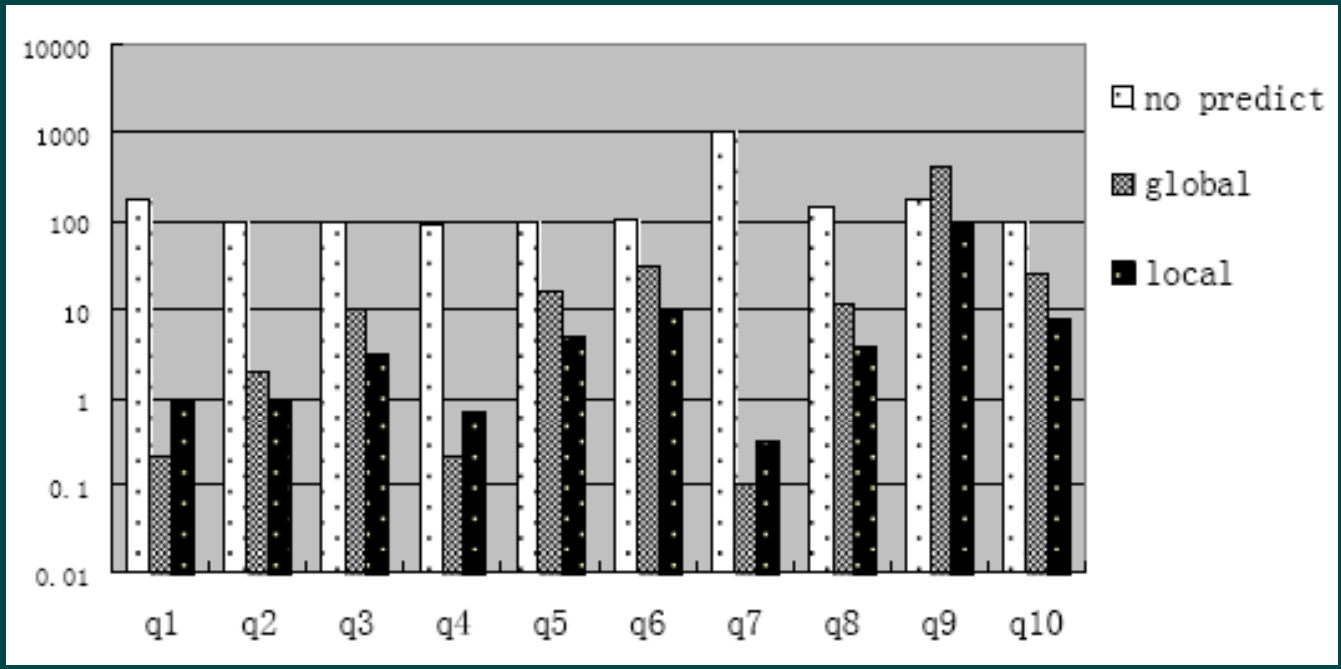
Pre-execution strategies

- **Local**
  - **Each node executes and keeps the result separately**
- **Global**
  - **There’s a result merging step after separately execution, keeping the result centralized.**





Pre-execution strategies

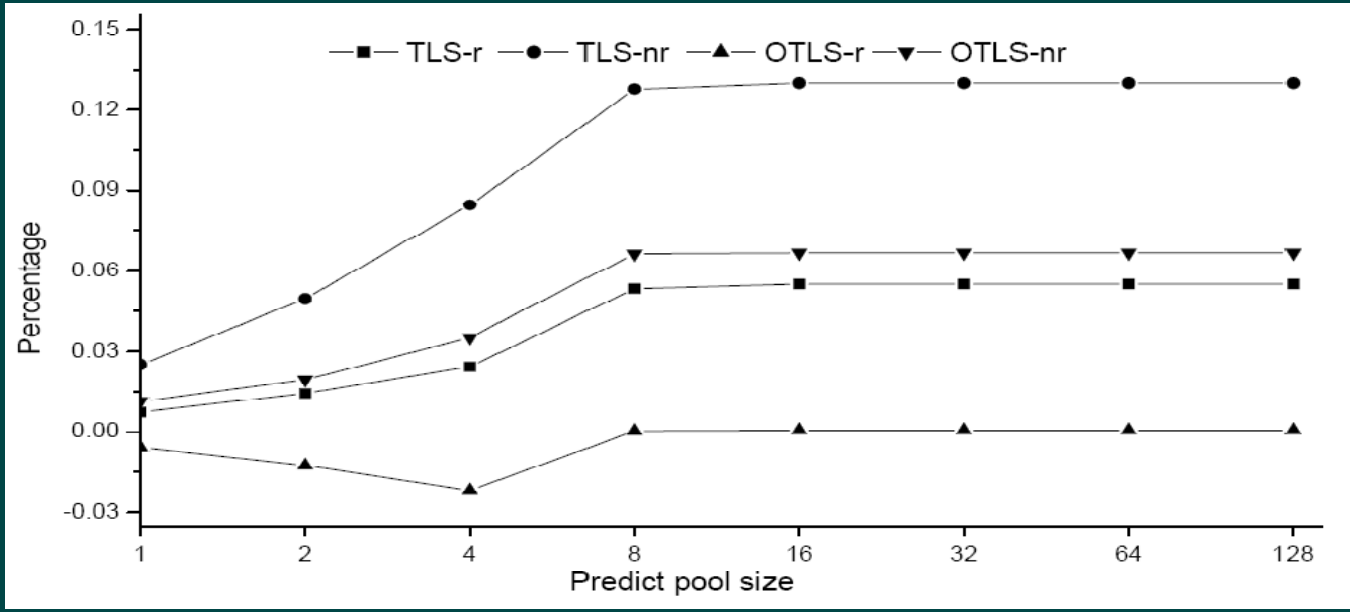


Query response time in different methods

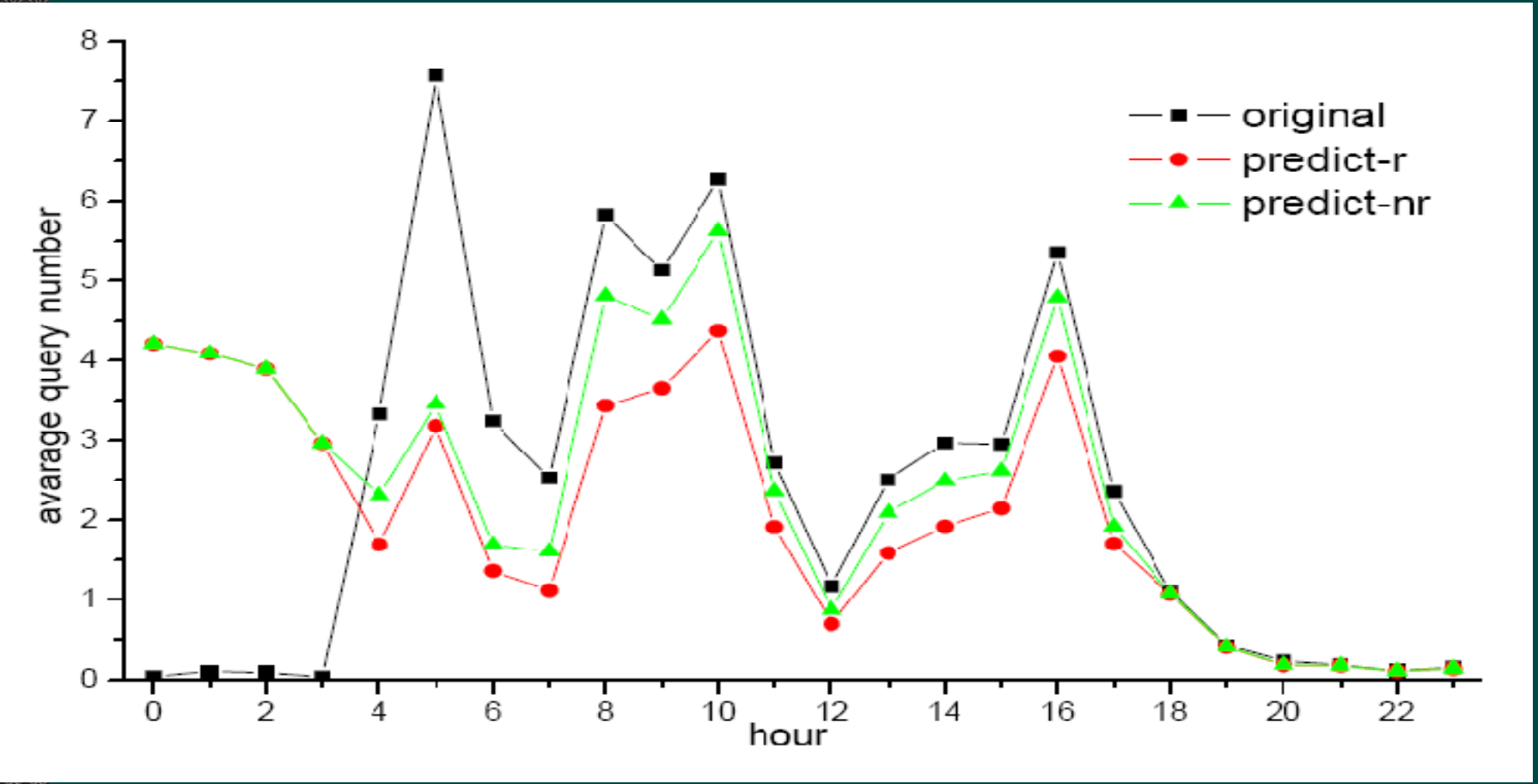
- Speculative execution can dramatically reduce response time, because of a greatly reduction of large amount of I/O cost.
- When speculative execution result size is small (as in q1, q4 and q7), the global method is better
- When the speculative execution result size is too large (as in q9), then global method may be even worse than no prediction method

Pre-execution Optimization

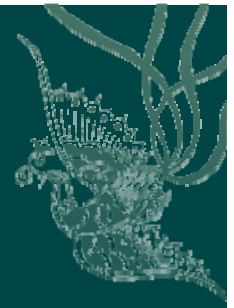
- **Predicates merge**
  - **Using ‘or’ : eg. t1=v1 or t1=v2 or …**
  - **Using ‘in’ : eg. t1 in (v1,v2,v3…)**
- **Reuse of pre-execution result**
  - **Keeps result in temporary tables**
  - **Further reduce system overhead**



Result Reuse—DBroker Workload



System Cost Analysis



No Prediction

$$T_{query} = T_{comp} + \left(\sum_i T_{opi} / p_{inner} \right) / p_{pipeline} \quad (1)$$

Prediction

$$T_{query} = T_{comp} + \left((1-\alpha) \cdot \sum_i T_{opi} / p_{inner} \right) / p_{pipeline} \quad (2)$$

System Workload

$$\begin{aligned} W &= N_{predict} + N_{miss} \\ &= N_{op} \cdot \alpha / \varepsilon + N_{op} \cdot (1-\alpha) \\ &= N_{op} \cdot \left(\frac{\alpha(1-\varepsilon)}{\varepsilon} + 1 \right) \end{aligned} \quad (3)$$

Extra Workload

$$W_{extra} = W - N_{op} = N_{op} \cdot \alpha \cdot (1-\varepsilon)/\varepsilon \quad (4)$$

SW with RR

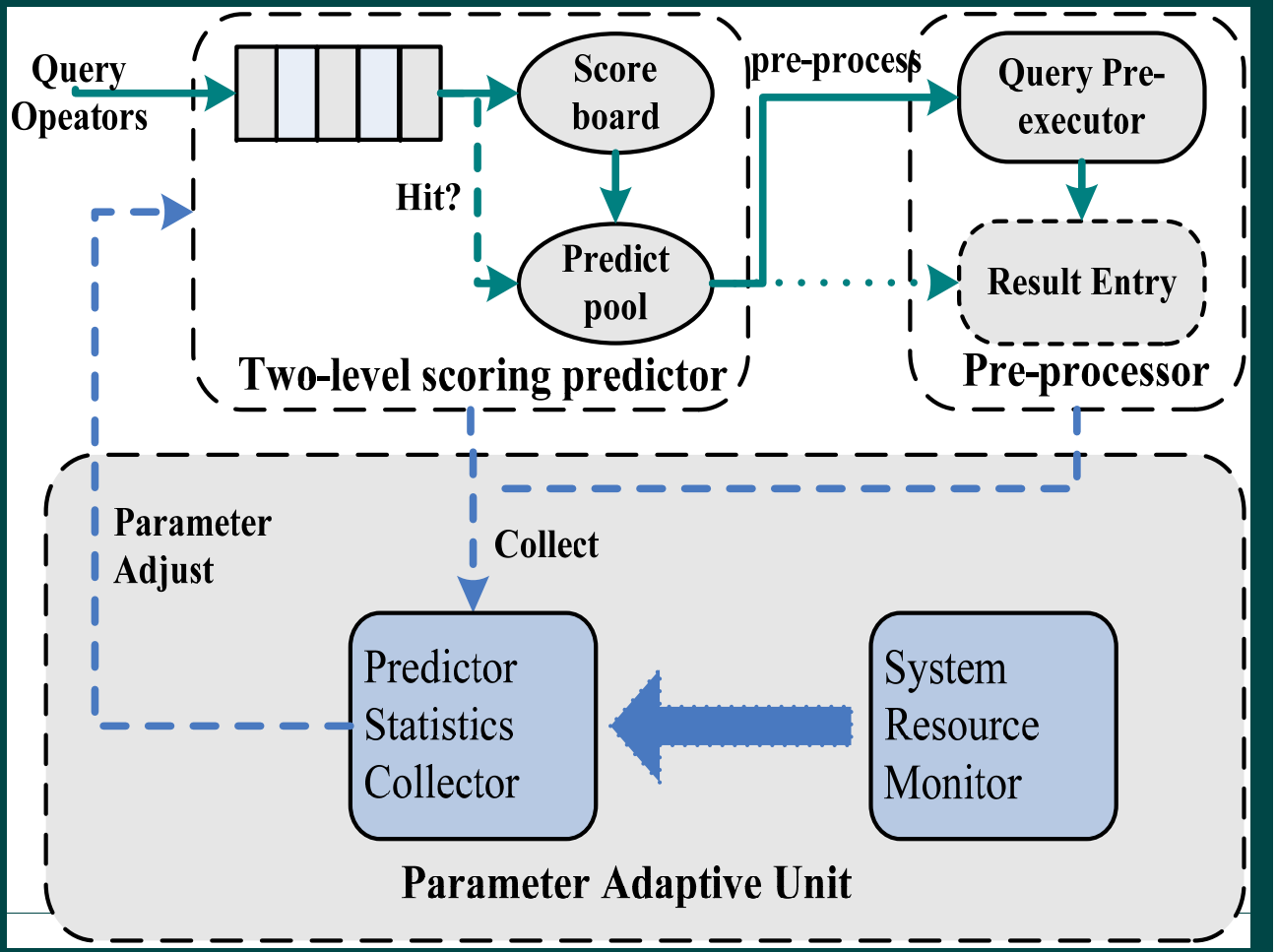
$$\begin{aligned} W &= N_{predict} + N_{miss} \\ &= N_{op} \cdot \alpha(1-r)/\varepsilon + N_{op} \cdot (1-\alpha) \\ &= N_{op} \cdot \left(\frac{\alpha(1-r-\varepsilon)}{\varepsilon} + 1 \right) \end{aligned} \quad (5)$$

EW with RR

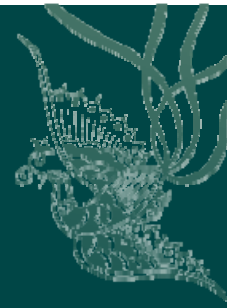
$$W_{extra} = N_{op} \cdot \alpha \cdot (1-r-\varepsilon)/\varepsilon \quad (6)$$

Adaptive Mechanism

- Prediction Stats Unit
  - Acc rate
  - Eff rate
  - Pre-exec time
  - DS size
- Resource Monitor Unit
  - CPU
  - MEM
  - DISK
  - NET



Summary & conclusions



- Discover attributes value locality of event stream data
- Prediction model and scoring algorithm
  - TLS and OTLS
  - Effects：accuracy rate >50%，extra workload <5%
- Local & Global pre-execution strategies
  - Pre-execution optimization
  - Effects：several times(10%-10) performance improvements while hitting
- The prediction model and techs can be used in other systems with hot-spot access characters
- Cannot：now only support table-scanning and filtering operations in the leaf nodes of the query execution tree

Thanks