Parallel Data Mining for Rules in Large Databases

Mohammed Javeed Zaki

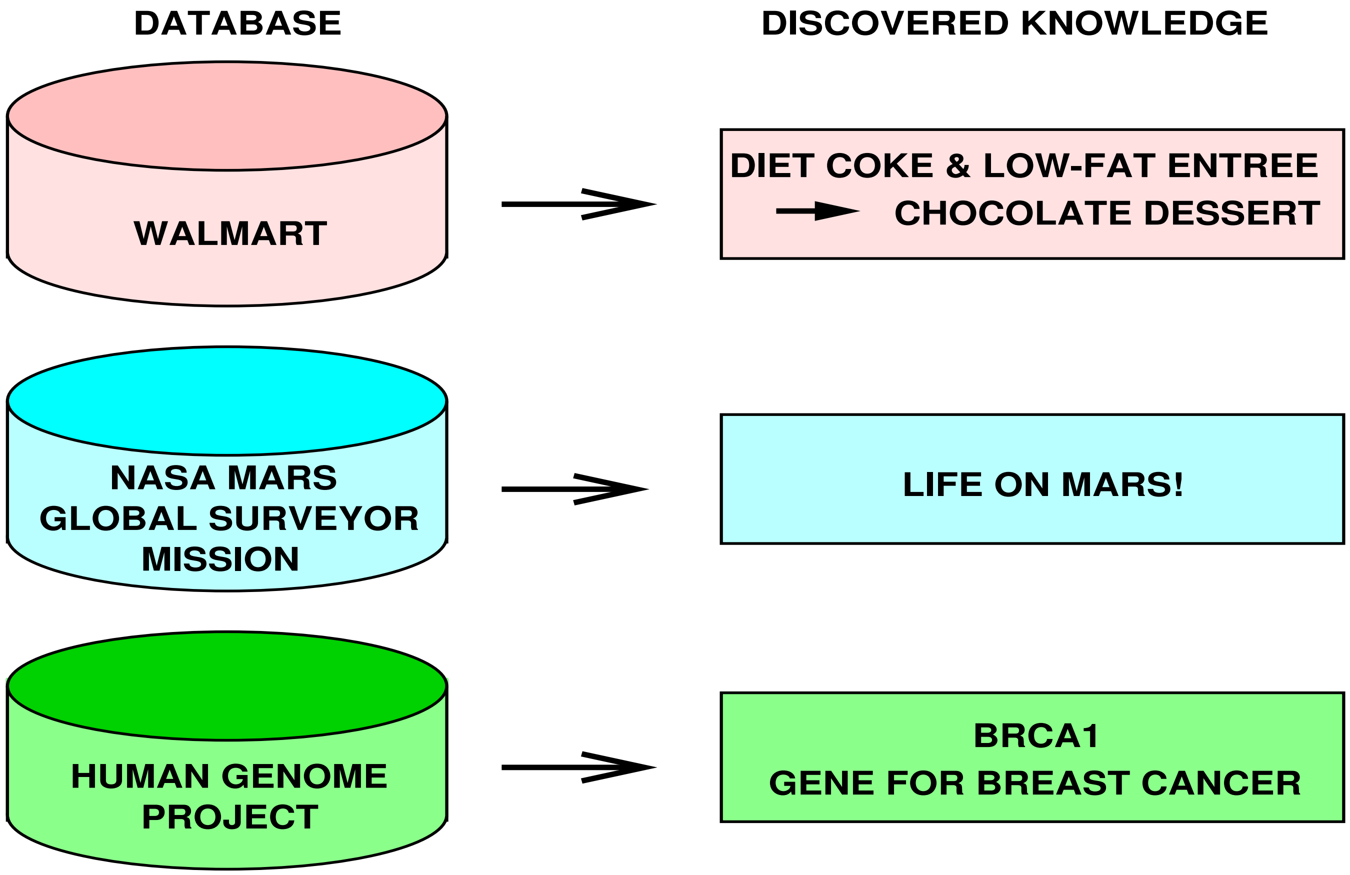
Computer Science Department

University of Rochester

Acknowledgments: Dr. Wei Li, Dr. Mitsu Ogihara, Srini Parthasarathy

Motivation

“We are drowning in information, but starving for knowledge” — John Naisbett



Mining Complexity

- Example Domains
  - Human Genome Project: GigaBytes of Genetic Code
  - NASA Mars Global Surveyor: Map Entire Martian Surface/Week
  - WWW: Millions of Sites, 100% Growth Rate/Year
  - WalMart: 20 Million Transactions/Day
- Efficient Algorithms: Exponential Search Space
- Parallel Computing: Scalability; Rapid Response

Talk Outline

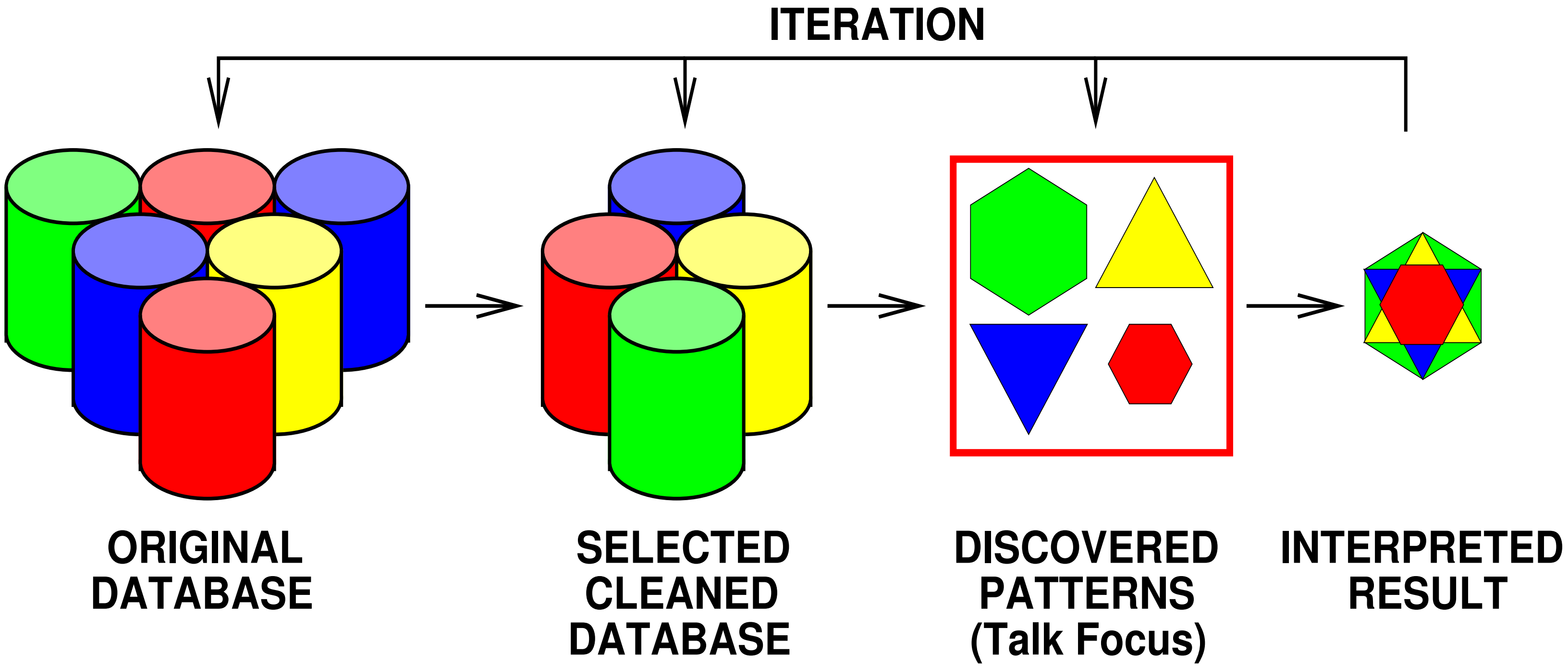
⇒ Data Mining Overview

- Association Discovery Overview
- Faster Association Algorithms
- New Parallel Algorithms
- Summary and Future Directions

What is Data Mining?

(Knowledge Discovery in Databases – KDD)

Data Mining is the process of automatic extraction of useful and understandable patterns in large databases



Data Mining Techniques

⇒ Association Discovery: Frequent Item-Sets

⇒ Parallel: Data Mining and KDD Journal, 1997

⇒ Sequential: 3rd Intl. Conference on KDD, 1997

- Parallel: 9th ACM SPAA Symposium, 1997
- Sampling: 7th Intl. Wkshop. Research Issues Data Engg., 1997
- Shared-Memory: Supercomputing 1996
- Memory Placement: URCS Technical Report, 1997

- Sequence Discovery: Frequent Item-Sequences
  - Faster Algorithm: URCS Technical Report, 1997
  - Application to Planning: URCS Technical Report, 1998
- Classification: Map Data Items to Predefined Classes
  - Shared-Memory: URCS/IBM Technical Report, 1998

Talk Outline

- Data Mining Overview

⇒ Association Discovery Overview

- Faster Association Algorithms
- New Parallel Algorithms
- Summary and Future Directions

What are Association Rules?

Given a set of Items, and a database of Transactions containing items, find Association Rules of the form:

IF {Item-Set1} THEN {Item-Set2}

provided that:

- {Item-Set1} and {Item-Set2} occur in a sufficient number of transactions — Minimum Support
- The rule has sufficient strength — Minimum Confidence

Discovery of Association Rules

- Find All Frequent Item-Sets with Minimum Support
  - Computationally Intensive: Exponential Search Space (2^m Possible Subsets, with m Items)
  - I/O Intensive: Multiple Database Accesses
- Generate All Association Rules with Minimum Confidence
  - Low I/O Overhead: Access Frequent Item-Sets and their Supports
- User Specifies Minimum Support and Confidence

Frequent Item-Sets: Example

DISTINCT DATABASE ITEMS

Jane Austen	Agatha Christie	Sir Arthur Conan Doyle	Mark Twain	P. G. Wodehouse
A	C	D	T	W

DATABASE

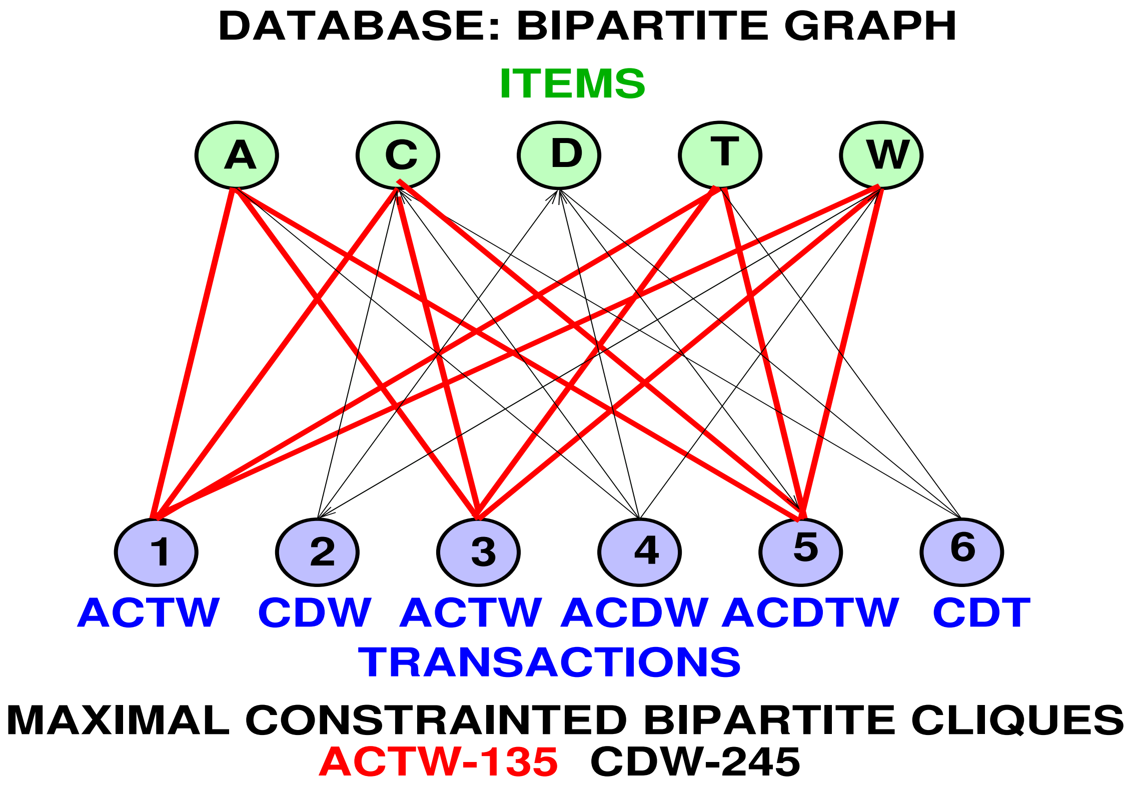
Transcation	Items
1	A C T W
2	C D W
3	A C T W
4	A C D W
5	A C D T W
6	C D T

ALL FREQUENT ITEM-SETS

Support	Item-Sets
100% (6)	C
83% (5)	W, CW
67% (4)	A, D, T, AC, AW CD, CT, ACW
50% (3)	AT, DW, TW, ACT, ATW CDW, CTW, ACTW
33% (2)	AD, DT, ACD, ADW CDT, ACDW
17% (1)	ADT, DTW, ACDT, ADTW CDTW, ACDTW

MAXIMAL ITEM-SETS (MIN SUPPORT = 50%): ACTW, CDW

Computational Complexity



- (Maximal) Frequent Item-set Discovery: Enumerate all (Maximal) Constrained Bipartite Cliques (I, T), where $I \in$ Items, $T \in$ Transactions, and $|T| \geq$ Minimum Support

Computational Complexity

- Deciding if there is a Maximal Frequent Item-Set of Length K is NP-Complete
  - Balanced Bipartite Clique [Garey, Johnson, 1979]
- Bounded Transaction Length (Items/Transaction): Enumerate All Maximal Frequent Item-sets in Time Linear in Number of Transactions and Items
  - Sparse Bipartite Graphs: $a^3 2^{2a} n$ (n is number of vertices, and a is sparseness bound) [Eppstein, 1994]
  - Still need efficient practical algorithms (a can be 10-20)

Association Mining Research

- Original Formulation [Agrawal, Imielinski, Swami, 1993]
- OCD [Mannila, Toivonen, Verkamo, 1994]
- Apriori [Agrawal, Srikant, 1995]

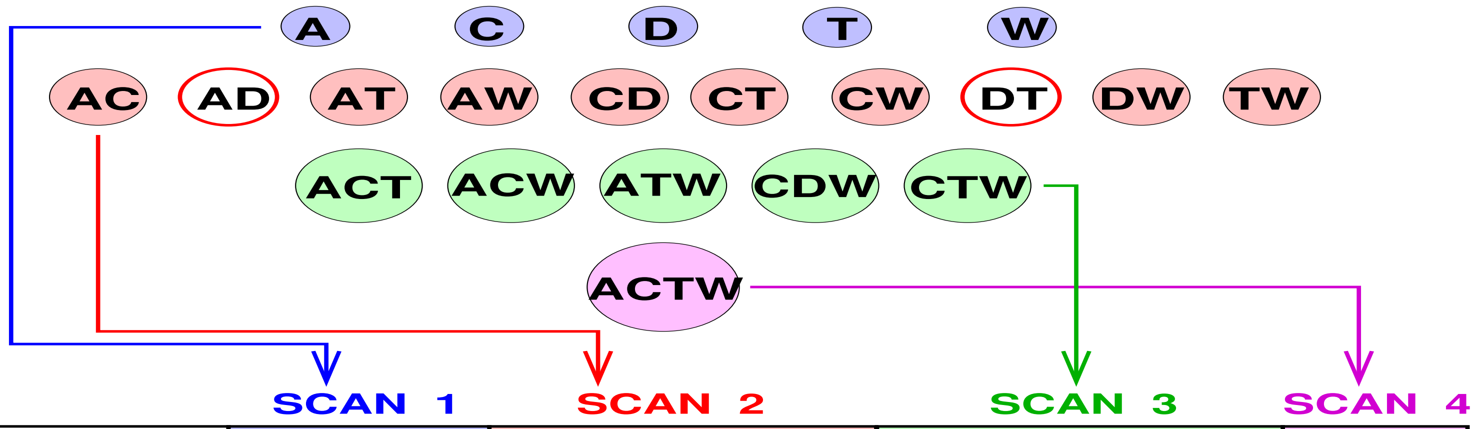
⇒ Apriori [Agrawal, Mannila, et al., 1996]

- DHP [Park, Chen, Yu, 1995]
- Partition [Savasere, Omiecinski, Navathe, 1995]
- Sampling [Toivonen, 1996]
- DIC [Brin, Motwani, Ullman, Tsur, 1997]

Apriori Algorithm

- Main Observation: All Subsets of Frequent Item-Set are Frequent
- Algorithm Sketch:
  - Separate Passes for Items, Pairs, Triples, ...
  - Construct Candidate Item-Sets for Next Pass
  - Prune Candidates with an Infrequent Subset
  - Scan Database to Count Remaining Candidates
  - Generate New Candidates from Frequent Item-Sets

Apriori Algorithm



Transcation	Items	Pairs	Triples	Quads
1	A C T W	AC AT AW CT CW TW	ACT ACW ATW CTW	ACTW
2	C D W	CD CW DW	CDW	---
3	A C T W	AC AT AW CT CW TW	ACT ACW ATW CTW	ACTW
4	A C D W	AC AD AW CD CW DW	ACW CDW	---
5	A C D T W	AC AD AT AW CD CT CW DT DW TW	ACT ACW ATW CDW CTW	ACTW
6	C D T	CD CT DT	---	---

SUPPORT COUNTING

Talk Outline

- Data Mining Overview
- Association Discovery Overview

⇒ Faster Association Algorithms

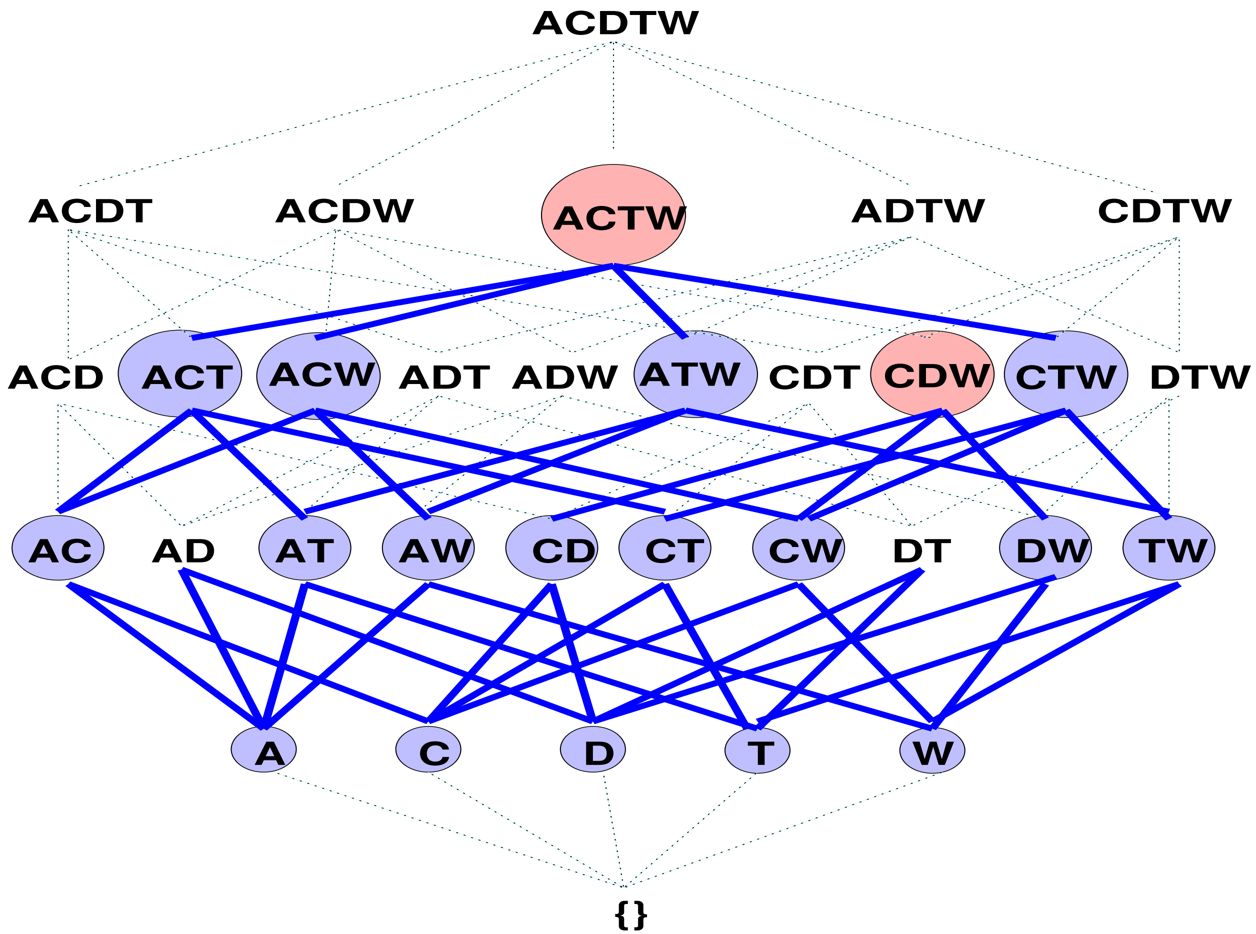
- New Parallel Algorithms
- Summary and Future Directions

The Eclat Algorithm

(Equivalence CLAss Transformation)

- Frequent Item-Set Lattice: Downward Closed on Support
- Vertical Database Format: Each Item-Set has a Tid-List — List of All Transactions Where it Occurs
- Support Counting: Simple Intersections of Tid-Lists
- Recursive Lattice Decomposition: Partition Original Problem into Several Smaller Sub-problems
- Independent Solution: Solve Each Sub-problem Independently in Main-Memory — Usually Only One Database Access!

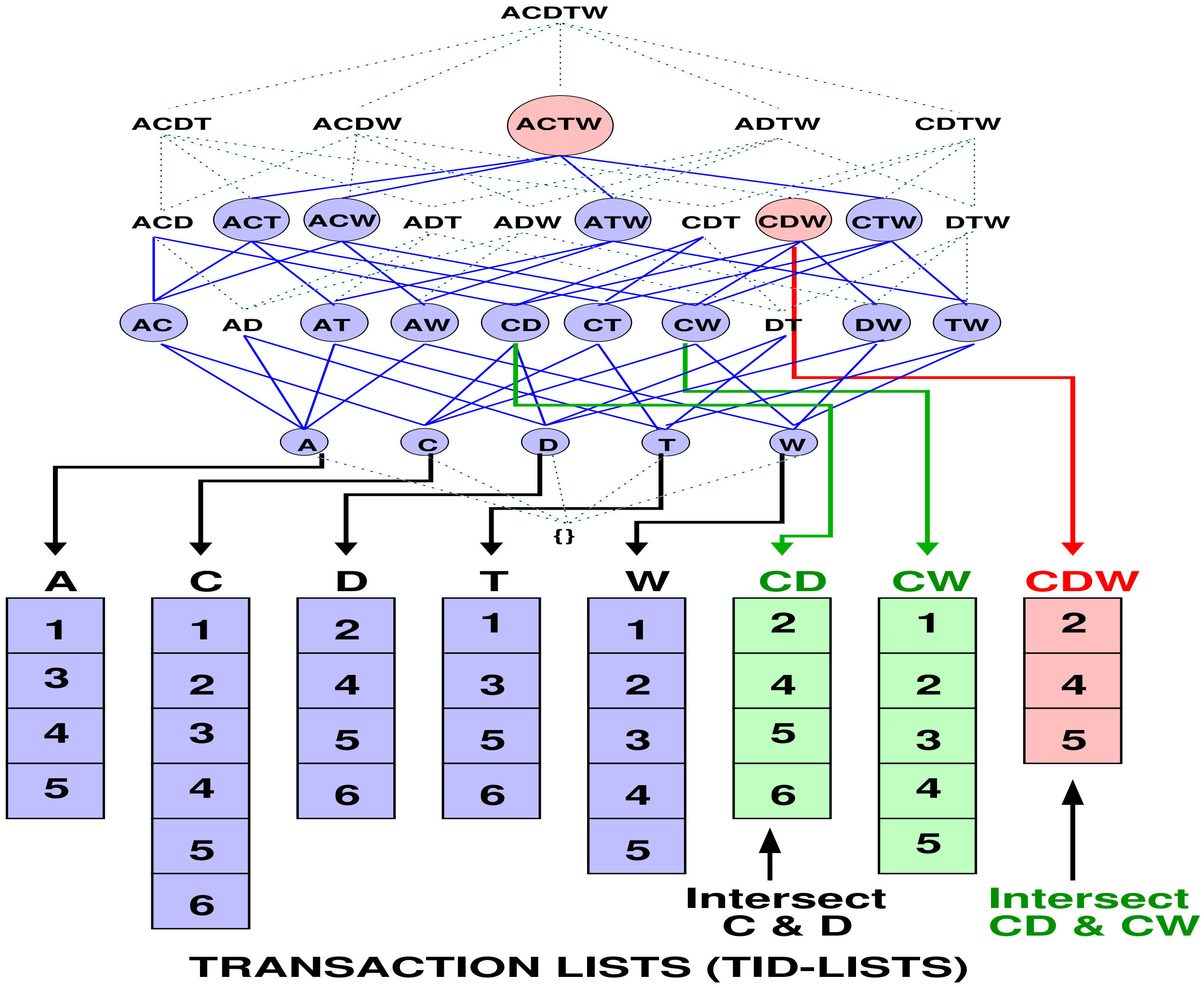
Eclat: Item-set Lattice



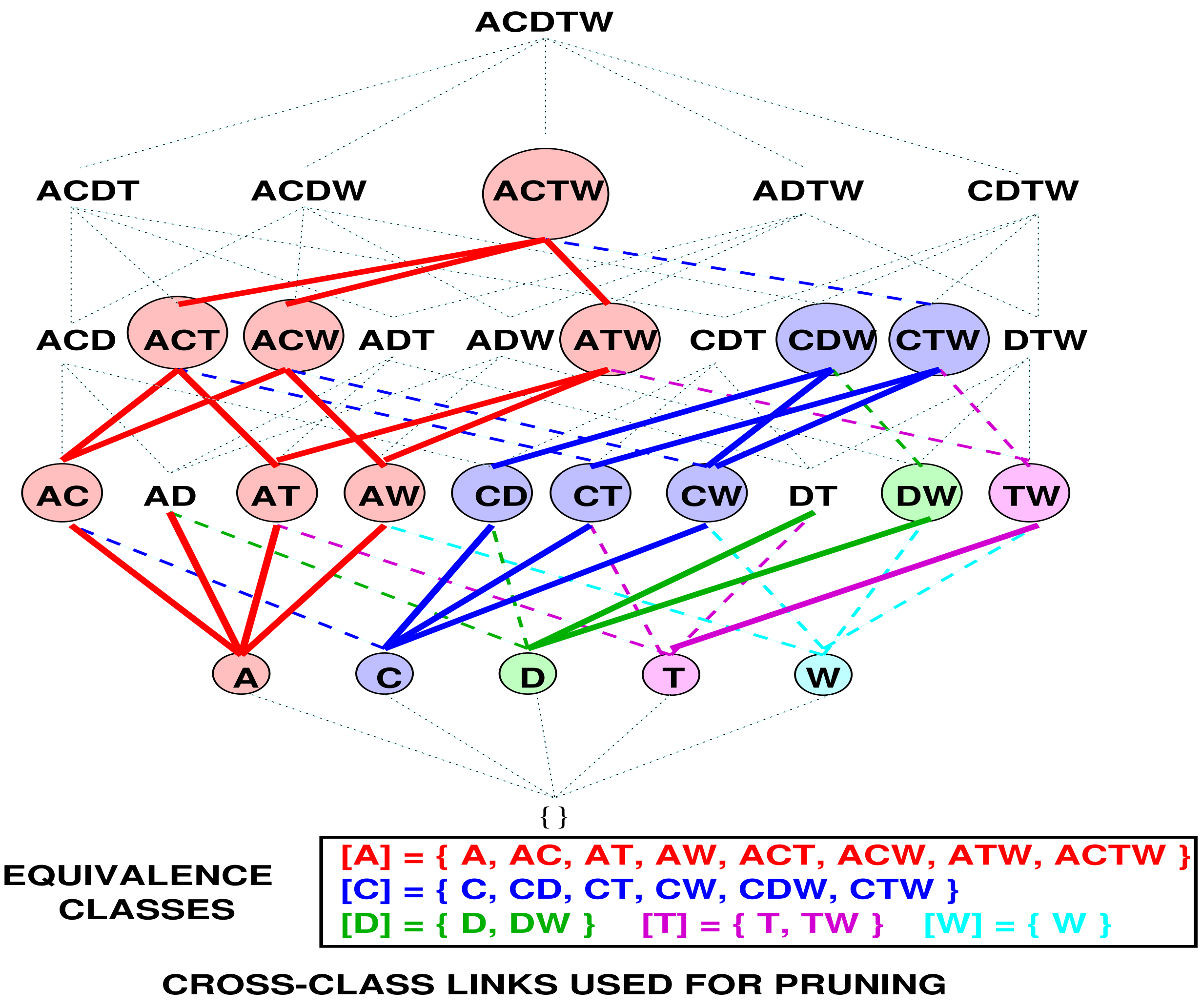
DOWNWARD CLOSED ON ITEM-SET SUPPORT

MAXIMAL FREQUENT ITEM-SETS: ACTW, CDW

Eclat: Support Counting



Eclat: Lattice Decomposition

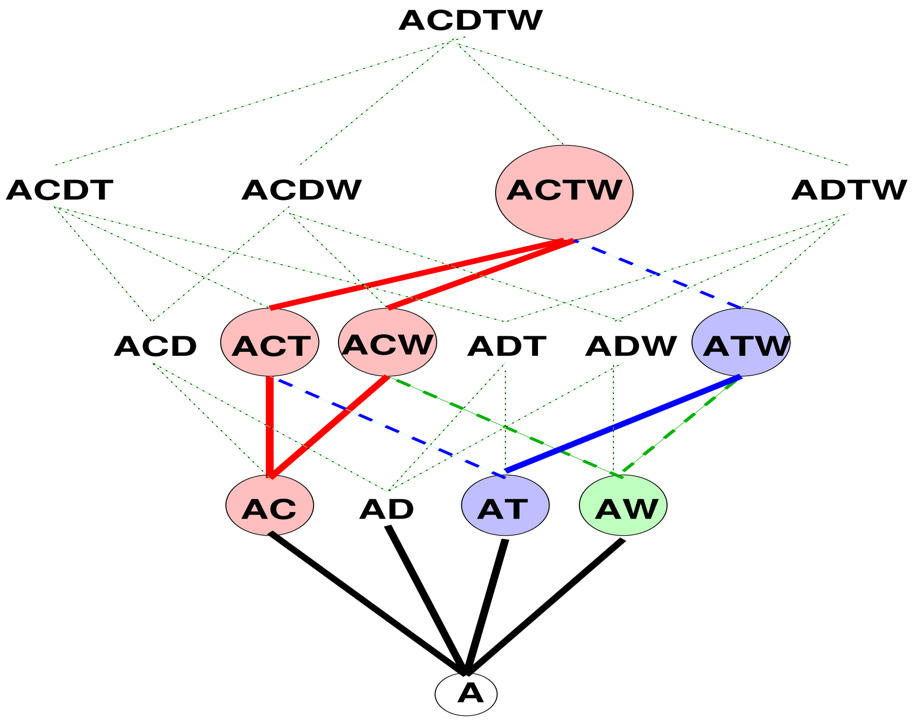


Lattice Search Strategies

- Bottom-Up Search
  - Start with Equivalence Classes of Item-Set Pairs
  - Generate Frequent Triples; Form New Classes
  - Repeat Until All Maximal Frequent Item-Sets Found
- Top-Down Search
  - Start with the Largest Lattice Element
  - If Frequent, Done! Else, Consider Each Subset
  - Repeat Until All Minimal Infrequent Item-Sets Found
- Hybrid Search
  - Quickly Determine a Maximal Frequent Item-Set
  - Bottom-Up Search on Remaining Item-Set Pairs

Bottom-Up Lattice Search

(Recursive Lattice Decomposition)

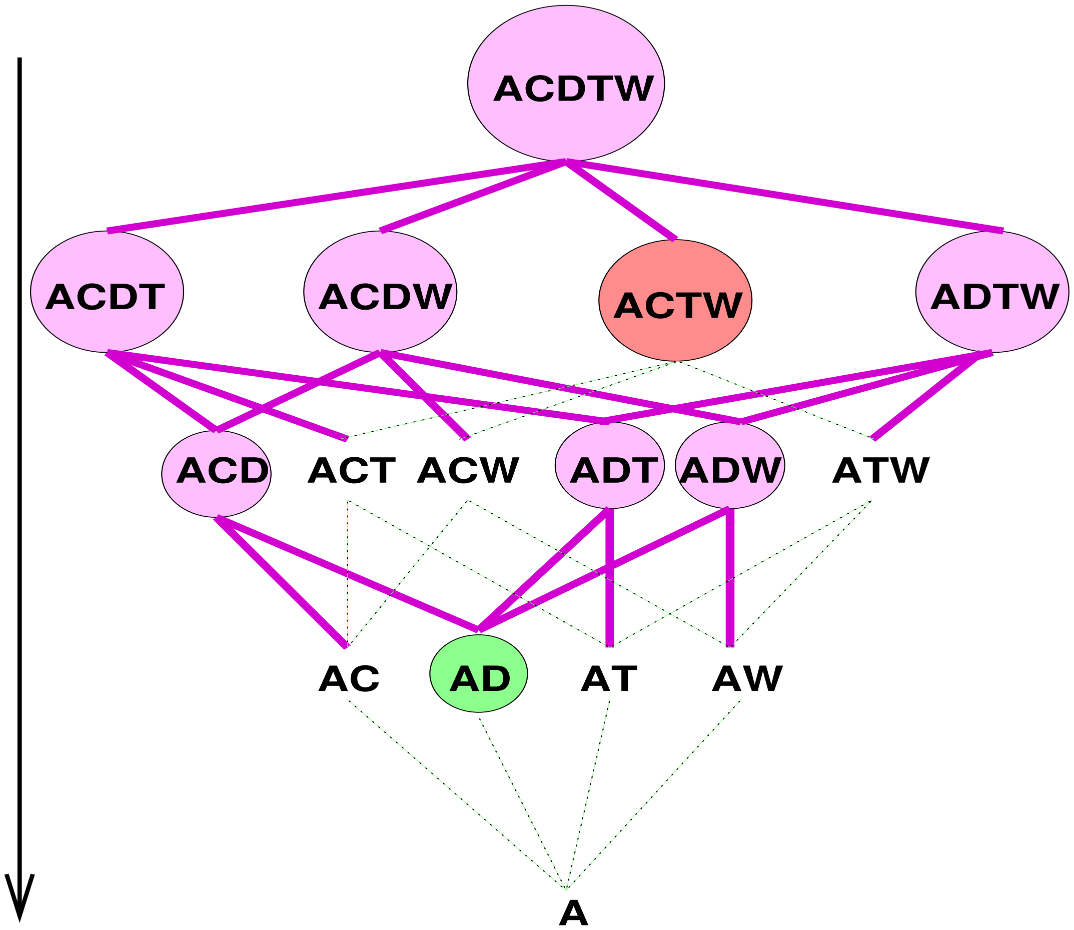


NEW EQUIVALENCE CLASSES

- [AC] = { AC, ACT, ACW, ACTW }
- [AT] = { AT, ATW }
- [AW] = { AW }

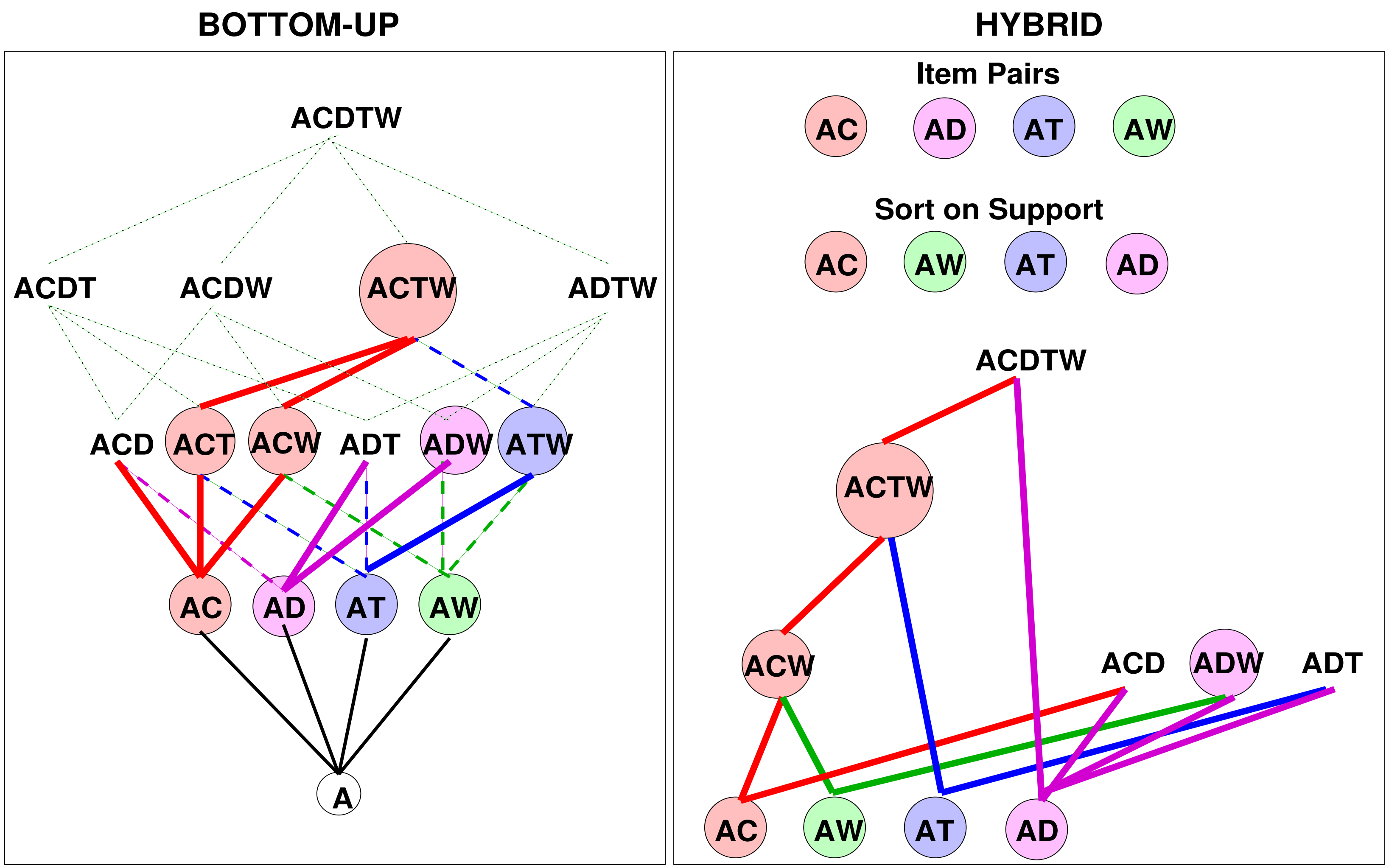
KNOWN MAXIMAL FREQUENT ITEMSETS: CDW, CTW

Top-Down Lattice Search

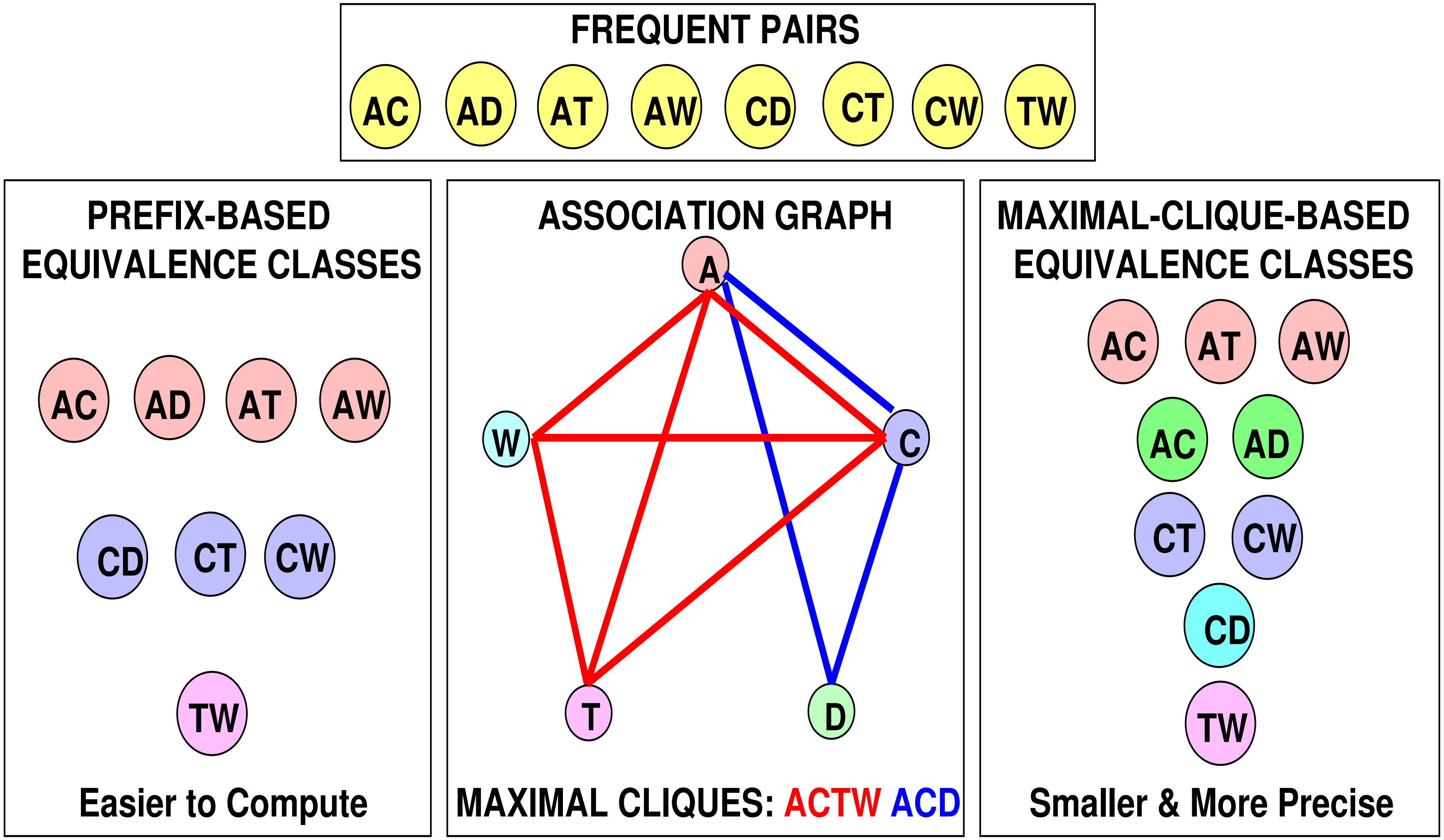


MINIMAL INFREQUENT ITEM-SET: AD
MAXIMAL FREQUENT ITEM-SET: ACTW

Hybrid Lattice Search



Can We Do Better? (Cliques)



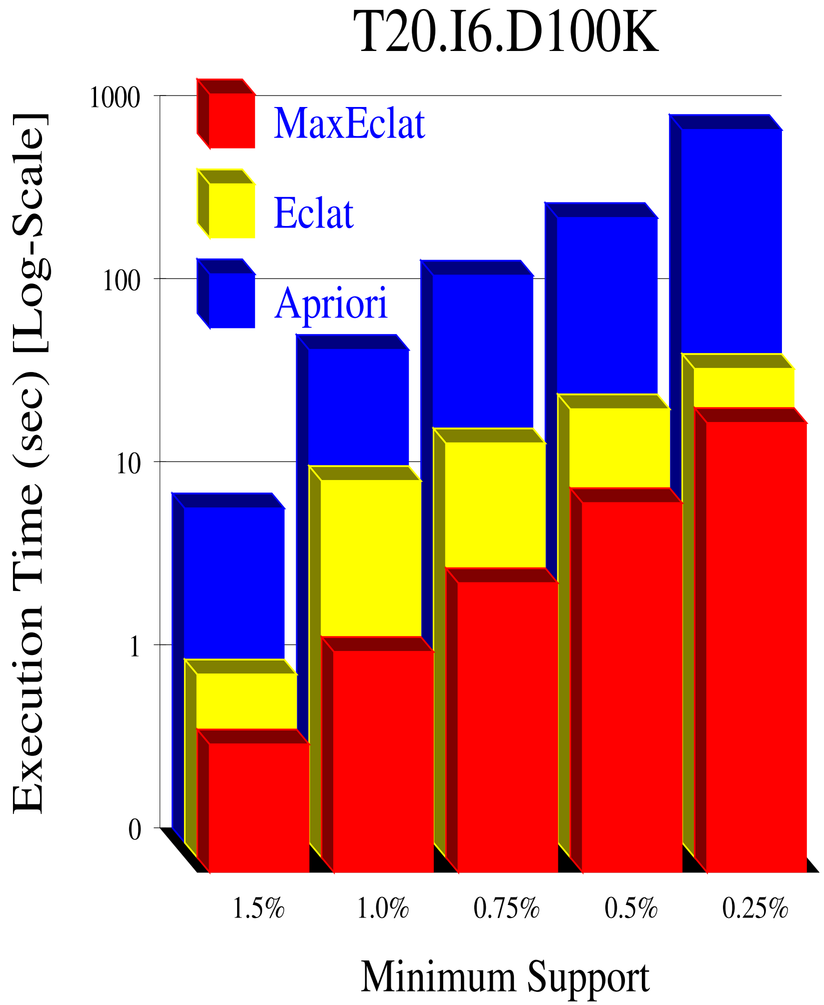
Computing Frequent Pairs

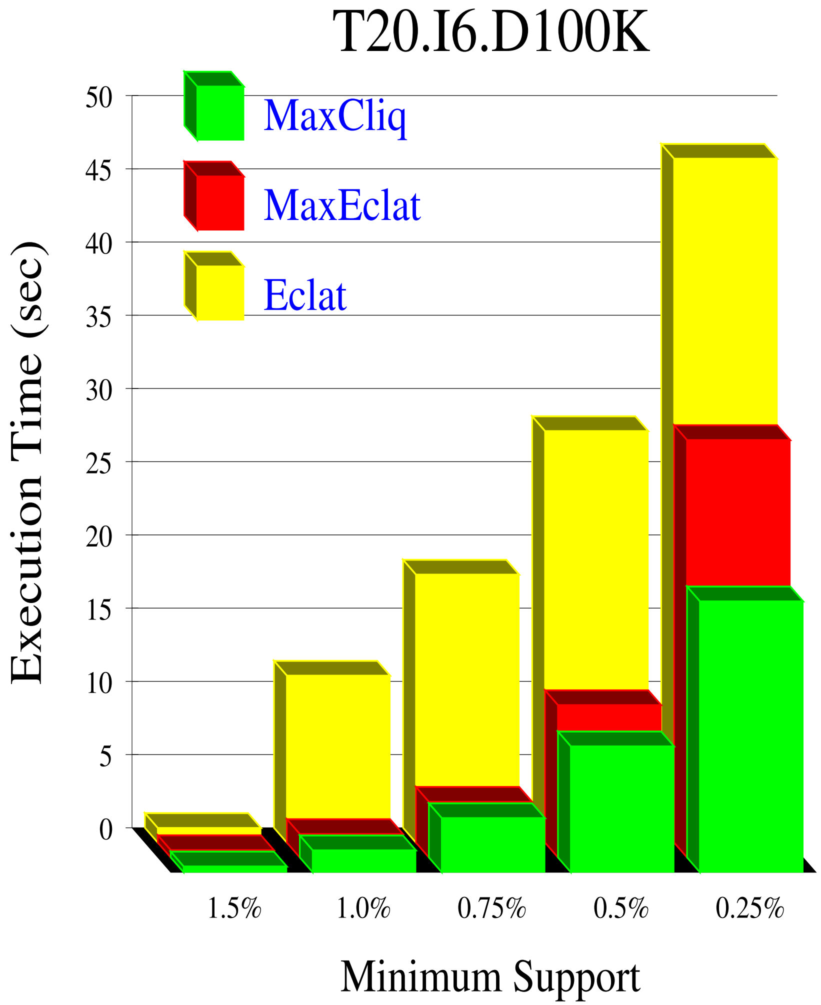
- Tid-List Database Format
  - Requires $O(N^2)$ Intersections for all Possible Pairs
  - Data Read: $A \cdot N(N-1)/2$ (A Avg. List Size; N Items)
- Transactional Database Format
  - Requires One Database Scan
  - Data Read: $A \cdot N$
  - Ratio: $1 : (N-1)/2$
- Solution:
  - Precompute Support of Pairs; Only Store Supports
  - Recover Transactional Format On-The-Fly

Experiments

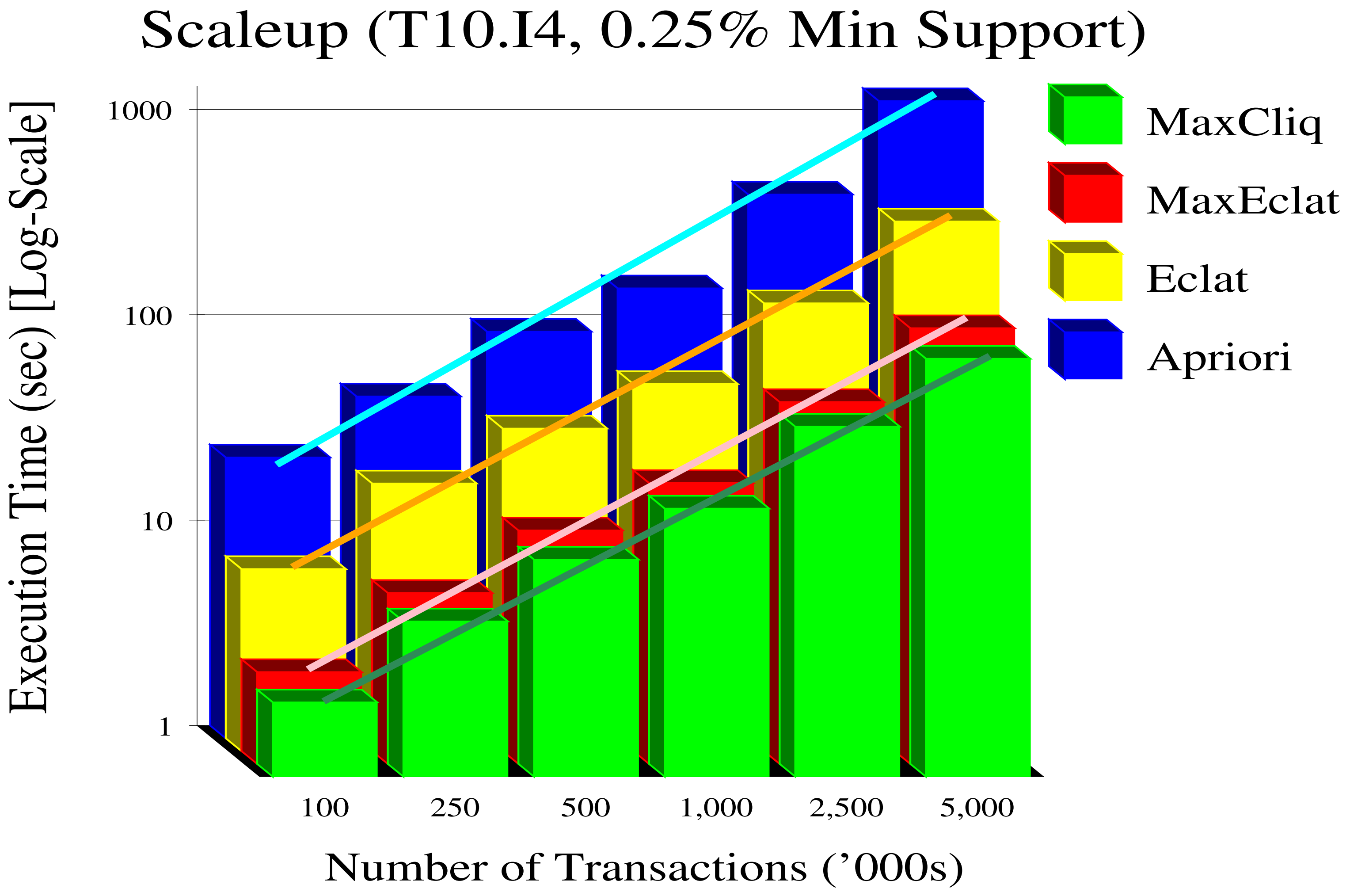
- Algorithms:
  - Eclat: Bottom-Up; Finds All Frequent Item-Sets
  - MaxEclat: Hybrid; Finds Maximal Frequent Item-Sets
  - MaxCliq: Clique-Based; Hybrid; Maximal
- Benchmark Database: T20.I6.D100K
  - Avg. Items per Transaction: 20
  - Avg. Maximal Frequent Item-Set Size: 6
  - Number of Items: 1,000
  - Number of Transactions: 100,000
- Machine: SGI
  - 100 MHz MIPS Processor
  - 256 MB Main Memory

Experiments





Experiments



Talk Outline

- Data Mining Overview
- Association Discovery Overview
- Faster Association Algorithms

$\Rightarrow$ New Parallel Algorithms

- Summary and Future Directions

Parallel Association Mining Research

- PDM [Park, Chen, Yu 1995]
- CCPD – Shared Memory [Zaki, Ogihara, et al., 1996]
- NPA, SPA, HPA [Shintani, Kitsuregawa, 1996]

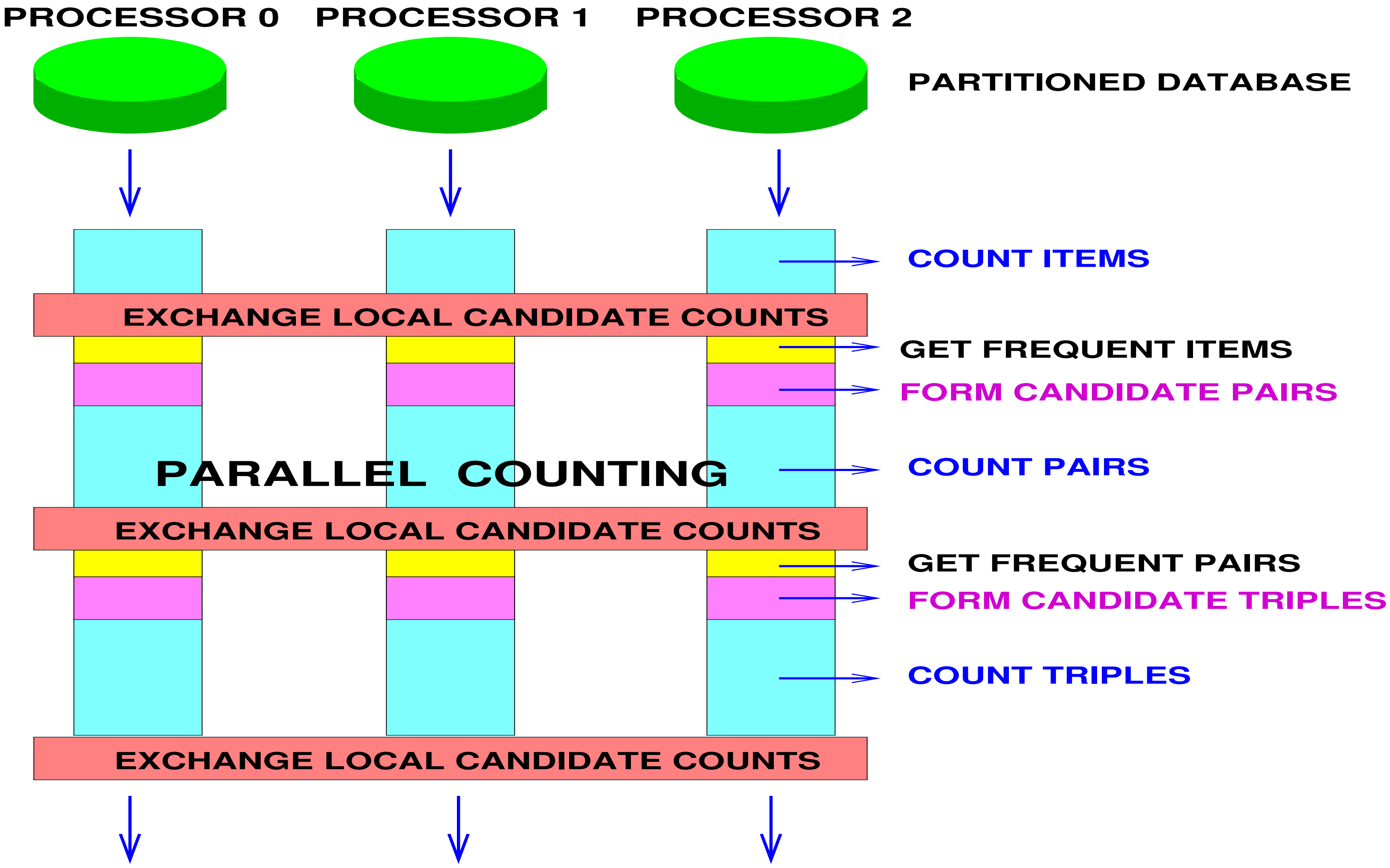
⇒ Count Distribution, Data Distribution, Candidate Distribution [Agrawal, Shafer, 1996]

- DMA [Cheung, Ng, Fu, Fu, 1996]
- Hybrid Distribution [Han, Karypis, Kumar, 1997]

Count Distribution

- Separate Passes for Item, Pairs, Triples, ...
- Database Partition: Equal-sized Blocks
- Candidate Generation: Replicated on Each Processor
- Support Counting: Scan Local Database for Local Counts
- Frequent Item-sets: Sum-reduction for Global Counts

Count Distribution



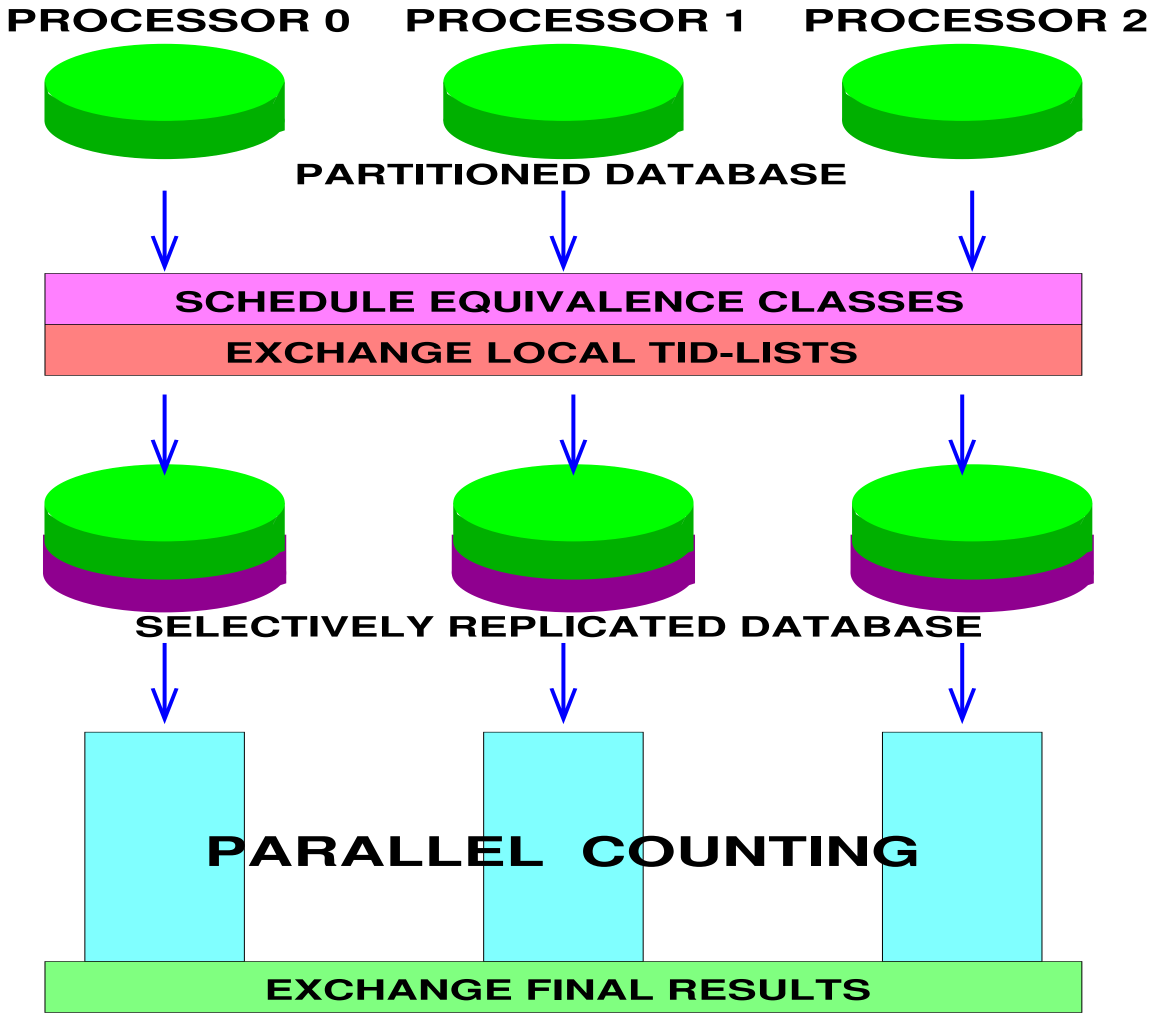
Count Distribution

- Advantages:
  - Simple to Implement
  - Minimizes Communication
- Disadvantages:
  - Repeated Database Scans
  - Doesn't Utilize Aggregate Memory
  - Synchronization per Iteration

Parallel Eclat Algorithm

- Database Partition: Split Tid-Lists among processors
- Lattice Decomposition: Form Equivalence Classes
- Class Schedule: Partition Classes among the processors
- Database Replication: Transmit relevant Tid-Lists
- Asynchronous Solution: Independently solve each class
  - Par-Eclat: Bottom-Up Search, All Frequent Item-Sets
  - Par-MaxEclat: Hybrid Search, Maximal Frequent Item-Sets

Parallel Eclat

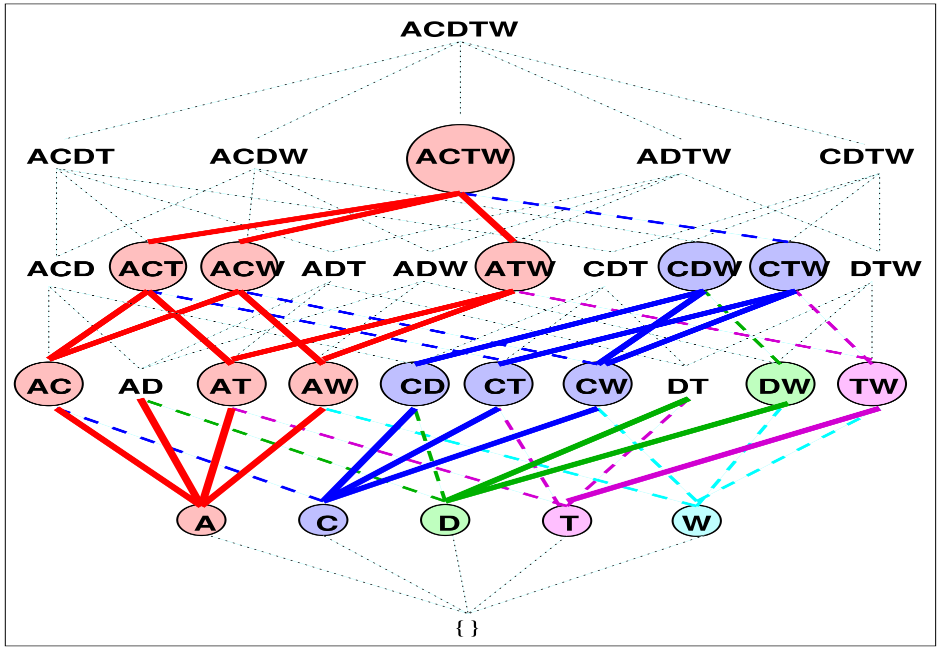


Class Scheduling: Opposing Criteria

- Maximize Load Balancing
  - Associate a weight with each class
  - Sort classes by weight
  - Greedy scheduling: Assign the next class to the processor with least total weight
- Minimize Database Replication (Communication)
  - Respect initial partitioning affinity
  - Schedule disjoint items on different processors
- Current Research Focus for Good Heuristics

Parallel Eclat Algorithm

ITEM SET LATTICE



CLASS PARTITIONING

EQUIVALENCE CLASSES

[A] = { AC, AT, AW }
[C] = { CD, CT, CW }
[D] = { DW }
[T] = { TW }

CLASS SCHEDULE

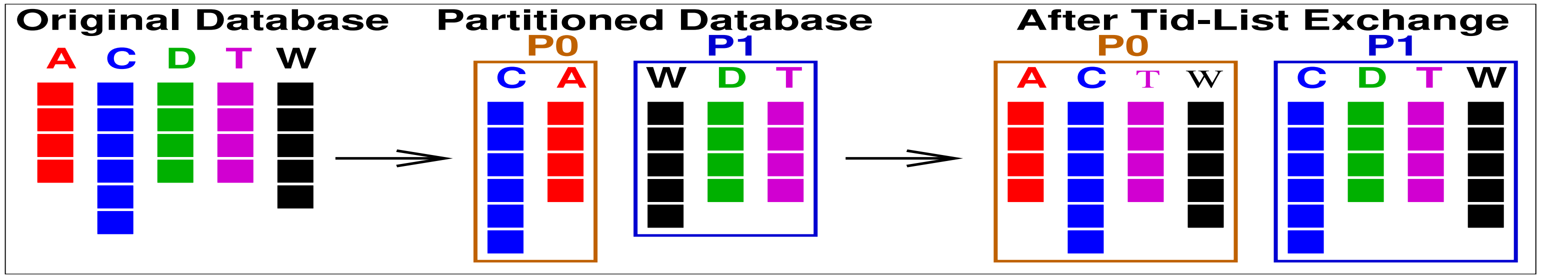
Processor 0 (P0)

[A] = { AC, AT, AW }
[T] = { TW }

Processor 1 (P1)

[C] = { CD, CT, CW }
[D] = { DW }

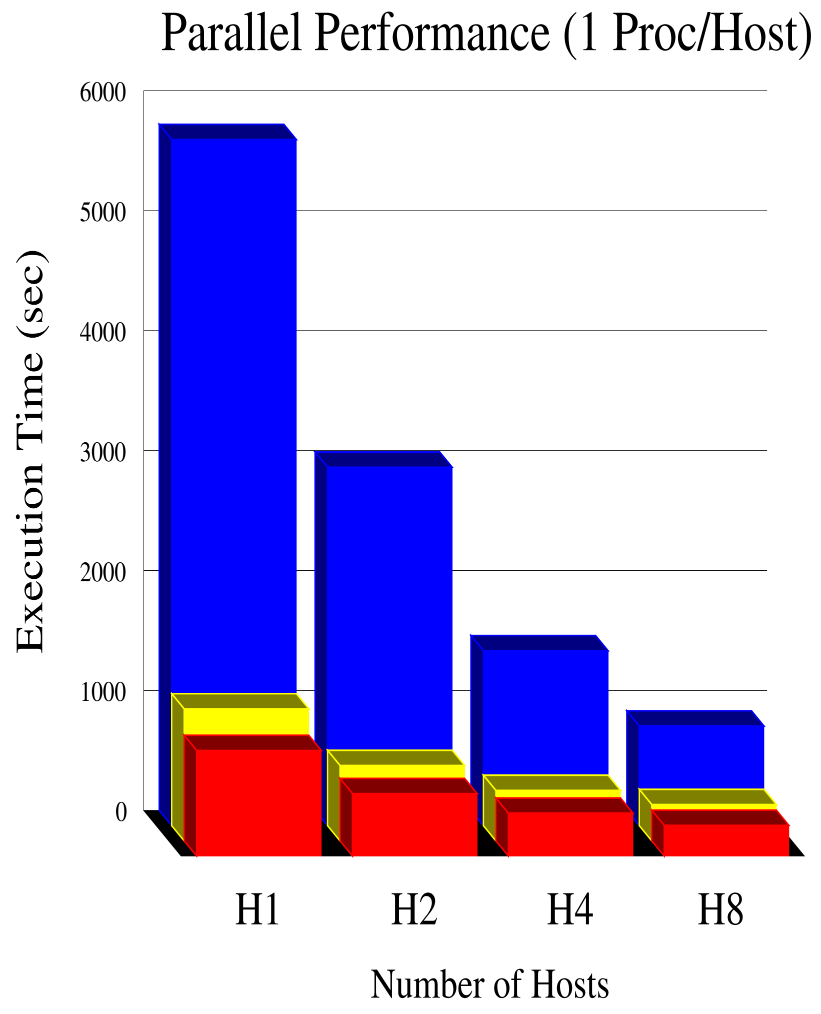
TID-LIST COMMUNICATION

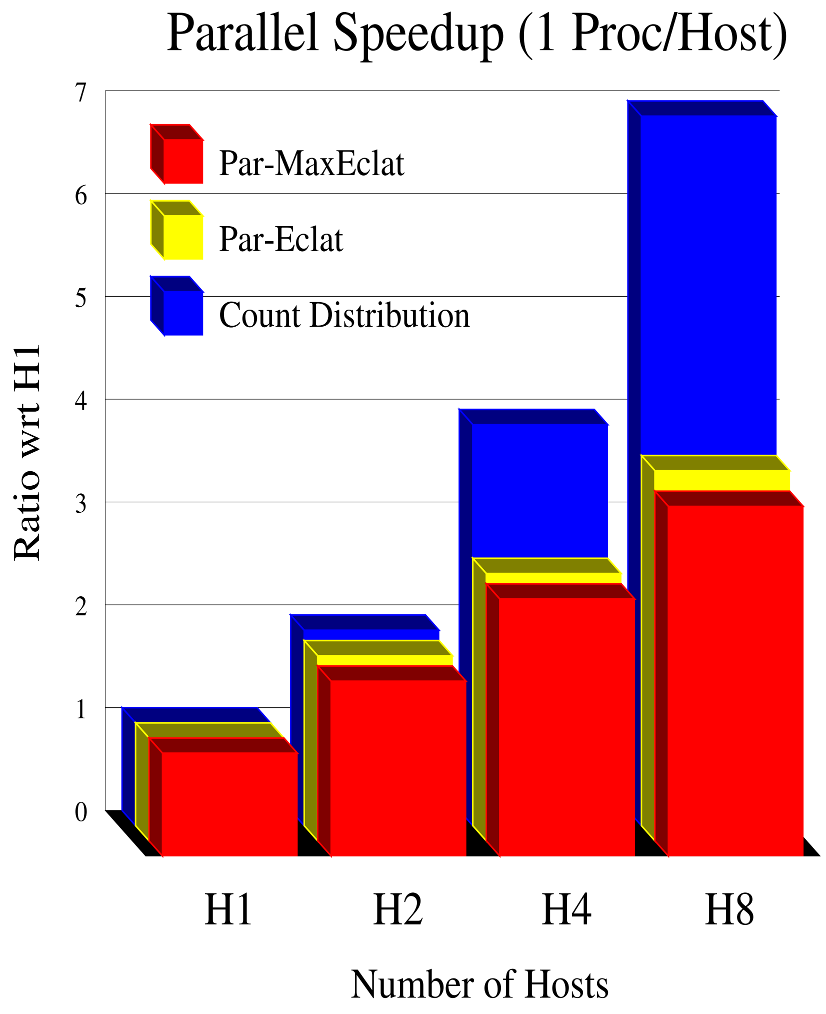


Parallel Experiments

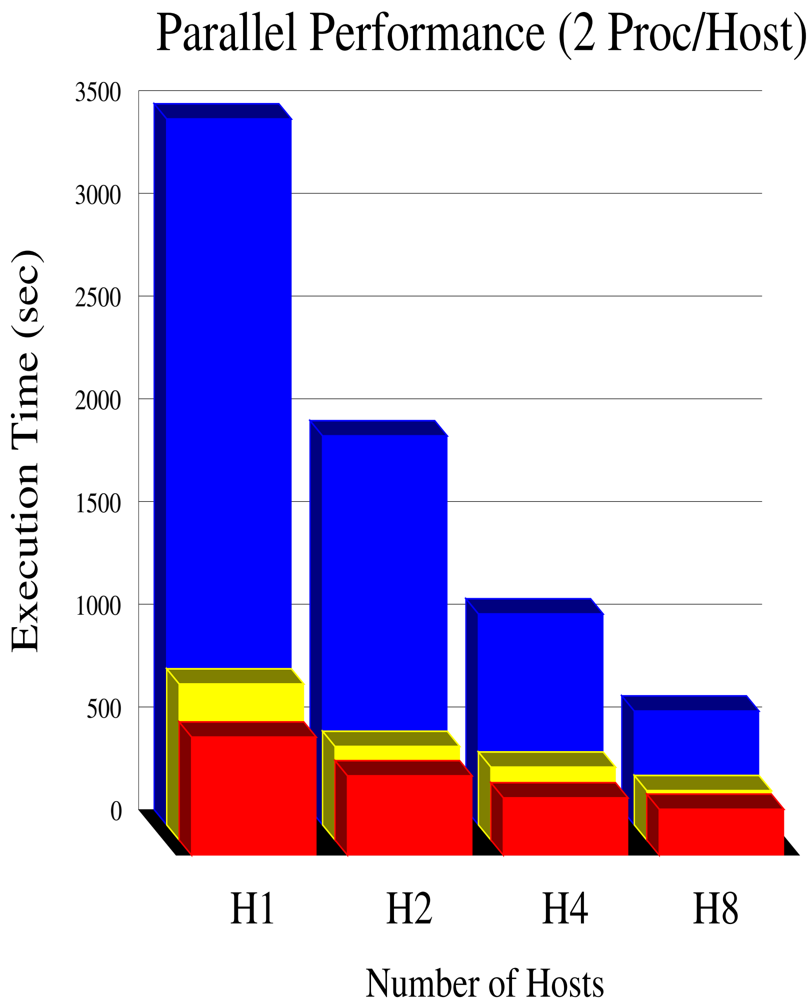
- Database: T20.I6.D4550K
  - Avg. Items per Transaction: 20
  - Avg. Maximal Frequent Item-Set Size: 6
  - Number of Items: 1,000
  - Number of Transactions: 4,550,000
  - Total Size: 370 MBytes
- Machine: 8 Digital Alpha 4-Processor SMP Cluster
  - 8 Hosts; 4 Processors per Host; 233 MHz Processor
  - 256 MB Main Memory; 1 MB L2 Cache
  - Memory Channel: Latency 5.2μs, Bandwidth 30MB/s
- Minimum Support: 0.25%
- Hierarchical Parallelization: SMP within Host, Message-Passing across Hosts

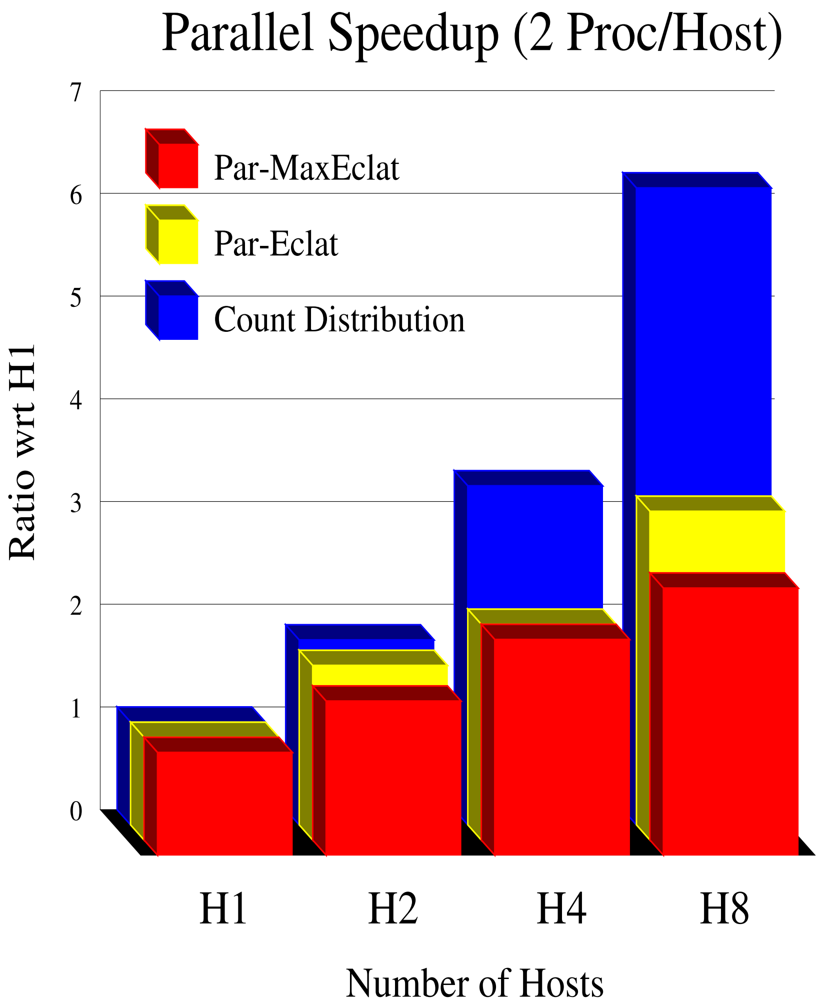
Parallel Experiments



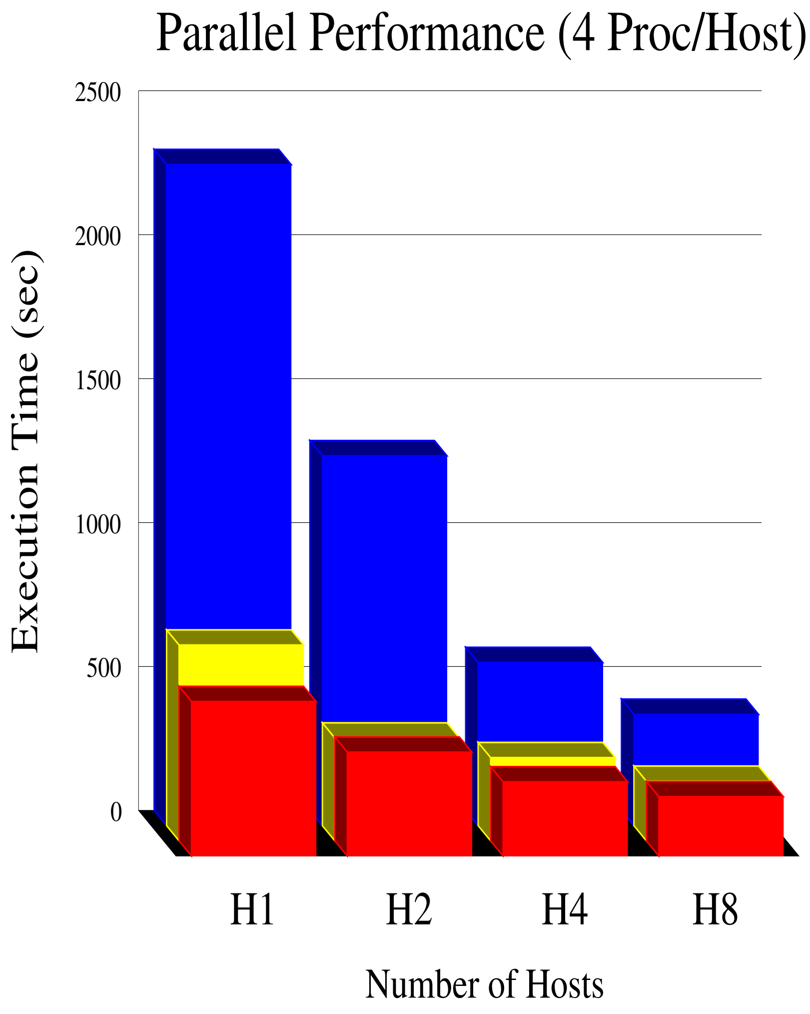


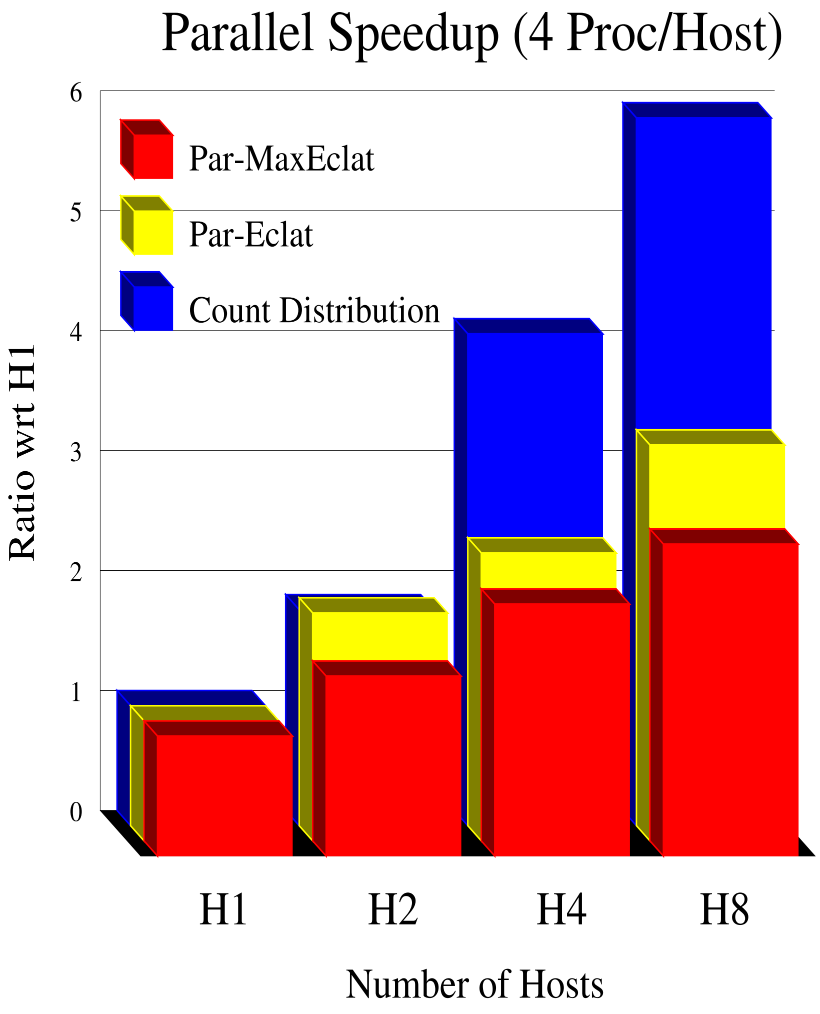
Parallel Experiments



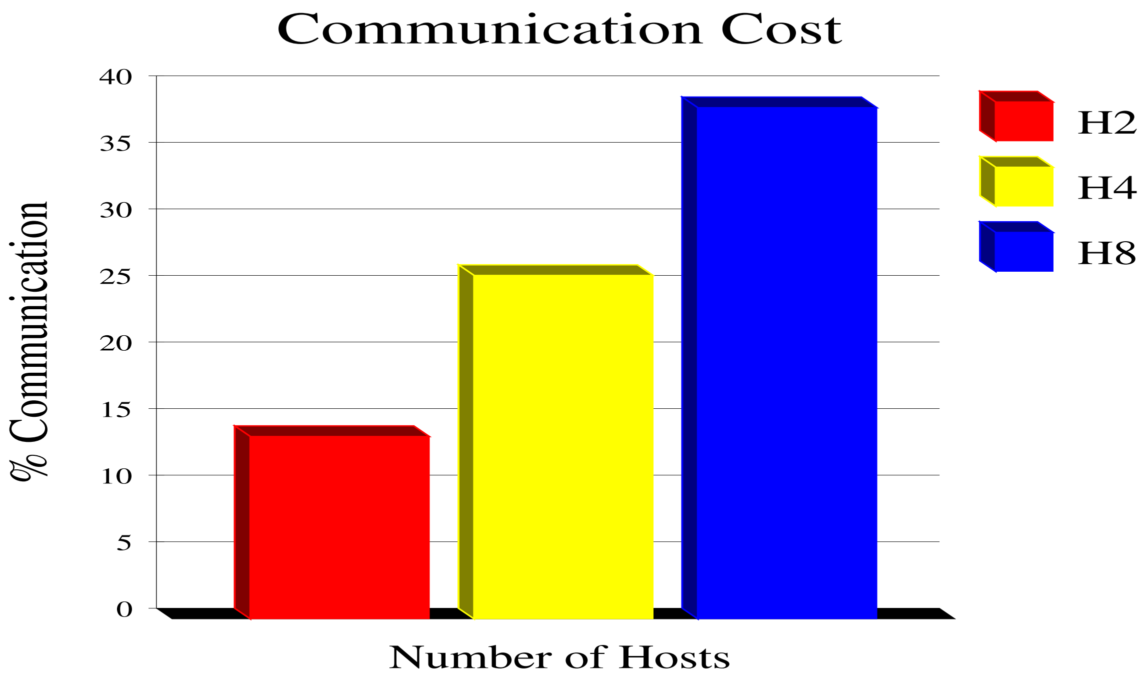


Parallel Experiments





Why Speedup Tapers Off?



- Computation vs Communication Ratio Reduces
- Communication Only Among Hosts
- Disk and Memory Contention With Increasing Processors/Host

Talk Outline

- Data Mining Overview
- Association Discovery Overview
- Faster Association Algorithms
- New Parallel Algorithms

$\Rightarrow$ Summary and Future Directions

Sequential Algorithm Summary

- Combinatorial Properties: Lattice Decomposition; Prefix and Clique-Based Equivalence Classes
- Efficient Search Techniques: Bottom-Up; Hybrid
- Item-set Enumeration: All or Maximal Frequent Item-sets
- Database Access: One Scan
- Efficient Counting: Simple Intersection Operations
- Performance: Order of Magnitude Improvements

Parallel Algorithm Summary

- Set-up Phase: Decouple Processors Up-front
- Asynchronous Phase: Enumerate Frequent Item-sets Independently
- Database Access: Two Scans — Replication and Frequent Item-set Enumeration
- Memory Usage: Utilize Aggregate Memory Effectively
- Good Locality: Tid-Lists (No Hash Structures)
- Performance: Factor of 2 to 5 Improvements

Future Directions

- Fast Algorithms: Different Data Mining Tasks
- Scalability: Number of Attributes and Records
- Incremental Mining: Fast Updation and Deletion
- Interactivity: Interrupt, Examine, and Guide Framework
- Data Locality and Type: WWW; Structured, Text, Image
- Parallel DBMS/File Systems: Tight Coupling
- Understandability: Too Many Discovered Patterns (e.g. AltaVista); Meta-Level Mining
- Applications: Molecular Biology, Performance Prediction