

# Plan Execution for Information Gathering

Craig Knoblock  
University of Southern California

This talk is based in part  
on slides from Greg Barish

Craig Knoblock

University of Southern California

1



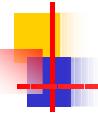
## Outline of talk

- Introduction
- Streaming dataflow execution systems
- A streaming dataflow plan language
- Optimizing execution of streaming dataflow plans
  - Streaming operators
  - Tuple-level adaptivity
  - Partial results for blocking operators
  - Speculative execution
- Discussion

Craig Knoblock

University of Southern California

2



## Motivation

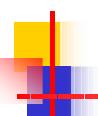
---

- Problem
  - Information gathering may involve accessing and integrating data from many sources
  - Total time to execute these plans may be large
- Why?
  - Unpredictable network latencies
  - Varying remote source capabilities
  - Thus, execution is often I/O-bound
- Complicating factor: **binding patterns**
  - During execution, many sources cannot be queried until a previous source query has been answered

Craig Knoblock

University of Southern California

3



## Traditional Approaches

---

- Executing information gathering plans
  - Generate a plan
  - Plan typically consists of a partial ordering of the operators
  - Execute the plan based on the given order
  - Operators process **all** of their input data before transmitting any results to consumer(s)
    - Operators as fast as their most latent input
  - Long delays due to the dependencies in the plan

Craig Knoblock

University of Southern California

4

# Streaming Dataflow Execution Systems

Craig Knoblock

University of Southern California

5

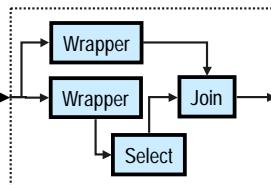
## Streaming Dataflow

- Plans consist of a network of operators
  - Each operator like a function
    - Example: Wrapper, Select, etc.
  - Operators produce and consume data
  - Operators “fire” when any part of any input data becomes available
  - Data routed between operators are relations
    - Zero or more tuples with one or more attributes

*Input*

City	State	Max Price
Santa Monica	CA	200000

*Plan*



*Output*

Address
100 Main St., Santa Monica, 90292
520 4th St. Santa Monica, 90292
2 Ocean Blvd, Venice, 90292

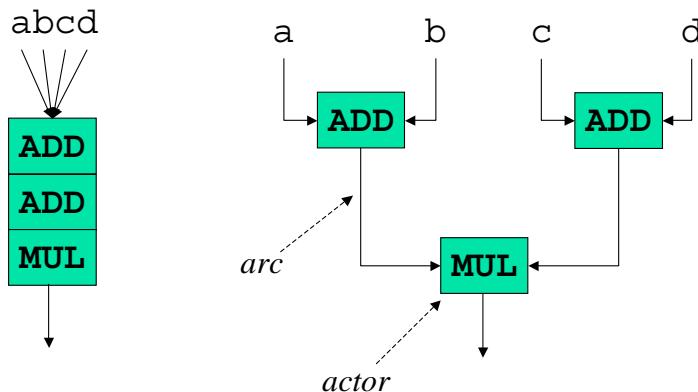
Craig Knoblock

University of Southern California

6

## Dataflow vs Von-Neumann

$$((a + b) * (c + d))$$



Craig Knoblock

University of Southern California

7

## Parallelism of Streaming Dataflow

- Dataflow (horizontal parallelism)
  - Decentralized, independent operator execution
  - Enables "maximally parallel" operator execution
    - Also known as the "dataflow limit"
- Streaming/pipelining (vertical parallelism)
  - Producer emits tuples to consumer ASAP
    - Producer & consumer can process same relation simultaneously
  - Effective because information gathering latencies can be high – even at the tuple level
    - Data often "trickles" out of I/O-bound operators

Craig Knoblock

University of Southern California

8

## Example: The RepInfo Agent

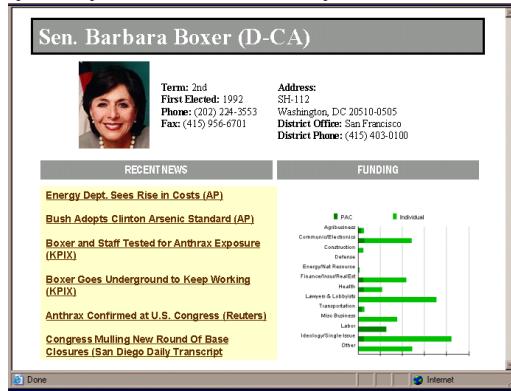
- INPUT

- Any street address

e.g., 4767 Admiralty Way, Marina del Rey, CA, 90292

- OUTPUT

- Federal reps
  - 2 senators,
  - 1 house member
- For each rep:
  - Recent news
  - Real-time funding information



Craig Knoblock

University of Southern California

9

## RepInfo Sources

Vote-Smart:  
-List of officials



University of Southern California

10

**RepInfo Sources**

**Vote-Smart:**  
-List of officials

**Yahoo**  
-Recent news

University of Southern California 11

**RepInfo Sources**

**Vote-Smart:**  
-List of officials

**Yahoo**  
-Recent news

**Open Secrets**  
-Funding graph

University of Southern California 12

**OpenSecrets – Navigation + Fetching!**

opensecrets.org Basics Who's Giving Who's Getting Get Local News and Issues HOME DONATE SEARCH

Election Overview | U.S. Congress | Congressional Committees | Political Parties | Presidential Data | Congressional Races

FIND A POLITICIAN:  
Search by Last Name:

Enter your Zip Code:

Who's got the most juice on Capitol Hill? Here's a list of the top industries contributing to members of the 107th Congress during the 2001-2002 election cycle. The first list shows the overall top 50 industries. The other two highlight the top 25 industries giving to members of each of the two major parties. In all cases, the Top Recipient listed is the individual member of the 107th Congress who received the most from the industry.

Totals shown here include only the money that went to current incumbents in Congress.

Top 50 Industries [2002]

Rank	Industry	Total	Dem Pct	GOP Pct	Top Recipient
1	Lawyers/Law Firms	\$8,383,065	68%	32%	Jean Carnahan (D-Mo)
2	Retired	\$4,944,324	41%	59%	Paul Wellstone (D-Minn)
3	Health Professionals	\$4,585,626	42%	57%	Greg Ganske (R-Iowa)
4	Real Estate	\$4,148,221	52%	48%	Charles E. Schumer (D-NY)
5	Securities/Invest	\$3,965,766	55%	45%	Charles E. Schumer (D-NY)
6	Insurance	\$3,797,370	38%	61%	Max Baucus (D-Mont)

Craig Knoblauch Done Internet

13

**OpenSecrets – Navigation + Fetching!**

opensecrets.org Basics Who's Giving Who's Getting Get Local News and Issues HOME DONATE SEARCH

Election Overview | U.S. Congress | Congressional Committees | Political Parties | Presidential Data | Congressional Races

Politicians featured on opensecrets.org:

- Jane Harman (D-Calif)

Campaign Finance Profiles

- 2002 (Member of Congress) ←
- 1996 (Member of Congress)
- 1994 (Member of Congress)

Race Profiles

- 2000 Race

You can use our search engine to find more references to "Harman" on opensecrets.org.

Craig Knoblauch http://www.opensecrets.org/politicians/summary.asp?CID=N00006750&cycle=2002 Internet

14

**OpenSecrets – Navigation + Fetching!**

Jane Harman: 2002 Politician Profile - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://www.opensecrets.org/politicians/summary.asp?CID=N00006750&cycle=2002

The opensecrets.org Basics Who's Giving Who's Getting Get Local News and Issues HOME DONATE SEARCH

Election Overview | U.S. Congress | Congressional Committees | Political Parties | Presidential Data | Congressional Races

2001-2002 Profile Total Raised Geographic Totals Sector Totals Top Industries Top Contributors

2001-2002 Data List PAC Contributions

Other Data Personal Finances 2000 Election 1995-96 Profile 1993-94 Profile (pdf file) Legislation (off-site)

States Home California Contribution Profile

Politicians Home

GO TO POLITICIAN (USE LAST NAME):  OK

**JANE HARMAN (D-CA)**

**Jane Harman**

2001-2002 Total Receipts: \$335,117  
2001-2002 Total Spent: \$146,838  
Cash on Hand: \$229,101  
First elected 2000

**Source of Funds:** (How to read this chart / methodology)

Source	Amount	Percentage
Individual contributions	\$131,990	(39.4%)
PAC contributions	\$202,985	(60.6%)
Candidate self-financing	\$0	(0.0%)
Other	\$142	(0.0%)

NOTE: All the numbers on this page are for the 2002 House election and based on data released electronically on January 14, 2003. The numbers do not add up..."

Feel free to distribute this material, but credit the Center for Responsive Politics.

**PAC Contribution Breakdown** (How to read this chart / methodology)

Source	Amount	Percentage
Business	\$139,770	(66.0%)
Labor	\$50,701	(23.9%)

Craig Knoblock Done Internet

15

**OpenSecrets – Navigation + Fetching!**

Jane Harman: 2002 Politician Profile - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://www.opensecrets.org/politicians/sector.asp?CID=N00006750&cycle=2002

The opensecrets.org Basics Who's Giving Who's Getting Get Local News and Issues HOME DONATE SEARCH

Election Overview | U.S. Congress | Congressional Committees | Political Parties | Presidential Data | Congressional Races

2001-2002 Profile Total Raised Geographic Totals Sector Totals Top Industries Top Contributors

2001-2002 Data List PAC Contributions

Other Data Personal Finances 2000 Election 1995-96 Profile 1993-94 Profile (pdf file) Legislation (off-site)

States Home California Contribution Profile

Politicians Home

GO TO POLITICIAN (USE LAST NAME):  OK

**JANE HARMAN (D-CA)**

**Contributions by Sector**

HOW TO READ THIS: The chart on this page shows the member's contributions from one of 13 main sectors within the business category, one for labor, one for ideological/single-issue, and one for "other."

More detailed breakdowns of these broad sectors can be found in the chart that follows.

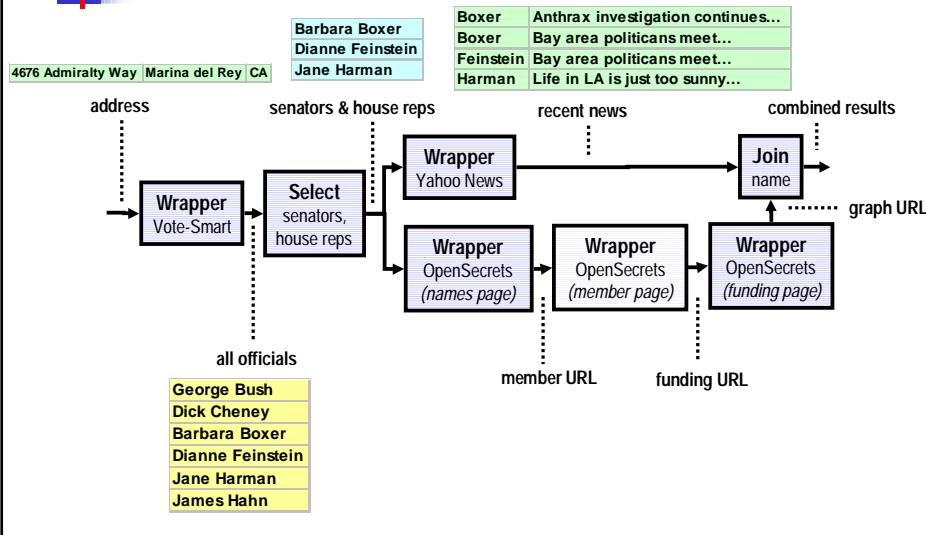
Historically, the finance industry has consistently been the biggest source of funds for elections. In 2000, financial interests gave \$133.9 million to federal candidates. This includes banks, insurance companies, the real estate industry, accountants and a variety of other financial professionals. Lawyers and lobbyists were third with \$100 million. The catch-all "miscellaneous business" category was fourth with \$75 million.

Sector	Total	PACs	Indivs
Agribusiness	\$10k	\$10k	\$10k
Communication/Electronics	\$10k	\$10k	\$10k
Construction	\$10k	\$10k	\$10k
Defense	\$10k	\$10k	\$10k
Energy/Nat Resource	\$10k	\$10k	\$10k
Finance/Insur/RealEst	\$10k	\$10k	\$10k
Health	\$10k	\$10k	\$10k
Lawyers & Lobbyists	\$10k	\$10k	\$10k
Transportation	\$10k	\$10k	\$10k
Misc Business	\$10k	\$10k	\$10k
Labor	\$10k	\$10k	\$10k
Ideology/Single-Issue	\$10k	\$10k	\$10k
Other	\$10k	\$10k	\$10k

Craig Knoblock Done Internet

16

## RepInfo agent plan



Craig Knoblock

University of Southern California

17

## Streaming Dataflow Systems for Network Environments

- Focus
  - Autonomous data sources on the Internet
  - Unpredictable network latencies
- Network Query Engines
  - Build plans to support queries
    - Tukwila
    - Telegraph
    - Niagara
- Agent-based Execution System
  - Support a richer plan language
    - Theseus

Craig Knoblock

University of Southern California

18

# A Streaming Dataflow Plan Language

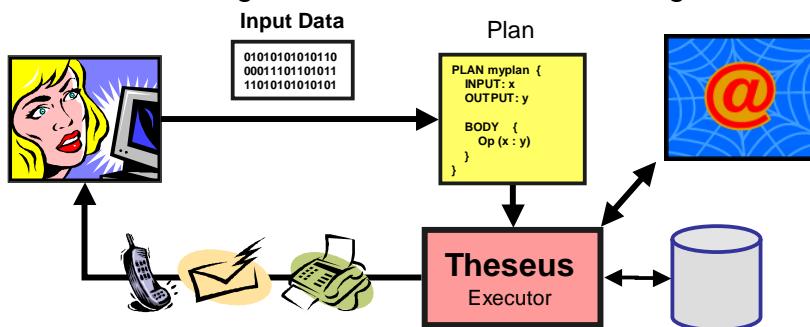
Craig Knoblock

University of Southern California

19

## Theseus

- A plan language and execution system for Web-based information integration
  - Expressive enough for monitoring a variety of sources
  - Efficient enough for near-real-time monitoring



Craig Knoblock

University of Southern California

20

## Expressivity

- Basic relational-style operators
  - **Select, Project, Join, Union**, etc.
- Operators for gathering Web data
  - **Wrapper**
    - Database-like access to a Web source
  - **Xquery, Rel2Xml, and Xml2Rel**
    - Enables better integration with XML sources
- Operators for monitoring Web data
  - **DbExport, DbQuery, DbAppend, DbUpdate**
    - Facilitates the tracking of online data
  - **Email, Phone, Fax**
    - Facilitates asynchronous notification

Craig Knoblock

University of Southern California

21

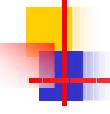
## Expressivity

- Operators for extensibility
  - **Apply**: single-row functions (e.g., UPPER)
  - **Aggregate**: multi-row functions (e.g., SUM)
- Operators for conditional plan execution
  - **Null**: Tests and routes data accordingly
- Subplans and recursion
  - Plans are named and have INPUT & OUTPUT
    - We can use them as operators (subplans) in other plans
  - Subplans make recursion possible
    - Makes it easy to follow arbitrarily long list of result pages that are each separated by a NEXT page link
  - Subplans encourage modularity & reuse

Craig Knoblock

University of Southern California

22



## Operators

```
operator (Input1, Input2,...:Output1, Output2,...)
    wait: waitInput1, waitInput2, ...
    enable: enableInput1, enableInput2, ...
```

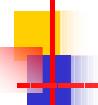
- Data formats

- Operators pass relations
- Relations are composed of tuples
- Each attribute of a tuple can be primitive, relation, or XML object

Craig Knoblock

University of Southern California

23



## Operator Streaming

- Operators support stream-oriented processing
  - Firing rule met when any input receives a tuple
    - This enables ASAP processing of data
  - End of data signaled by end-of-stream (EOS)
- Operators vary on when they can begin output:
  - Union: **immediately** (i.e., for each input)
  - Minus: **after EOS for second input has arrived**
  - Email: **after EOS for all inputs have arrived**

Craig Knoblock

University of Southern California

24

## Wrapper Operator

**PURPOSE:** Extract data from web pages as relation

- **INPUT:**

- `Name`: URL prefix of wrapper
- `bind_map`: Wrapper binding map
- `bind_dat`: Binding tuples

- **OUTPUT:**

- `new_rel`: Incoming relation joined with new attributes

```
auth = [USER  PASSWORD  
        greg   secret]  
  
wrapper("http://fetch.com?wrapper=foo",  
        "user=$user, pwd=$password", auth : quotes)  
  
quotes = [USER  PASSWORD  SYMBOL  PRICE  
          greg   secret     ORCL    15.50  
          greg   secret     CSCO    21.50]
```

Craig Knoblock

University of Southern California

25

## Plans and Subplans

```
plan planName  
{  
    input: planInput1, planInput2, ...  
    output: planOutput1, planOutput2, ...  
    body {  
        operator (opInput1,... : opOutput1,...)  
        operator ...  
        ...  
    }  
}
```

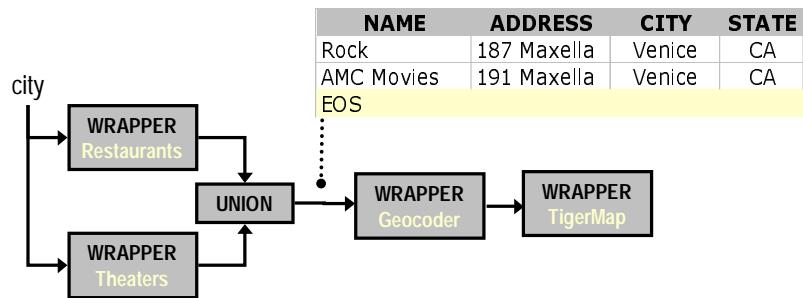
- Plans can be called just like operators (subplans)

Craig Knoblock

University of Southern California

26

## Example plan: TheaterLoc



Craig Knoblock

University of Southern California

27

## TheaterLoc Plan

```

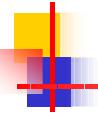
PLAN theaterloc
{
    INPUT: city
    OUTPUT: latlons, map_url

    BODY
    {
        wrapper ("cuisinenet", "name, addr", city : restaurants)
        wrapper ("yahoo_movies", "name, addr" city : theaters)
        union (restaurants, theaters : addresses)
        wrapper ("geocoder", "name, lat, lon", addresses : latlons)
        wrapper ("tigermap", latlons : map_url)
    }
}
    
```

Craig Knoblock

University of Southern California

28



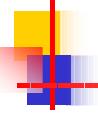
## Transactions

- Enable
  - Concurrent plan access by multiple clients
  - Recursive plan execution
- Transactions each assigned unique ID
- Individual transactions can be aborted
- All transactions are assigned a “time to live”
  - Unprocessed data is garbage collected by Theseus

Craig Knoblock

University of Southern California

29



## Conditionals and Recursion

- Conditional outputs are defined by enabling outputs depending on the action results

```
Null(inStream : outStreamTrue,outStreamFalse)
```

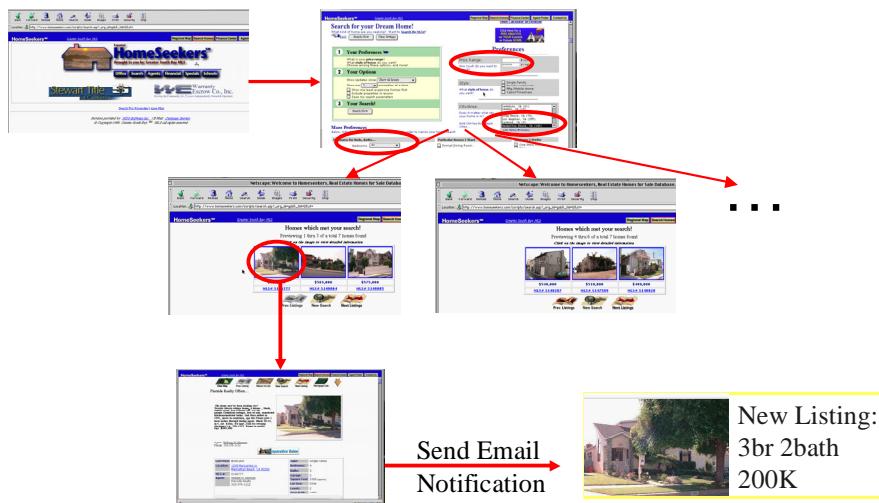
- Plans can be called recursively
  - Termination defined by conditional operators
  - Transactions support recursive calls in same execution environment
  - System provides tail-recursion optimization

Craig Knoblock

University of Southern California

30

## Real Estate Plan



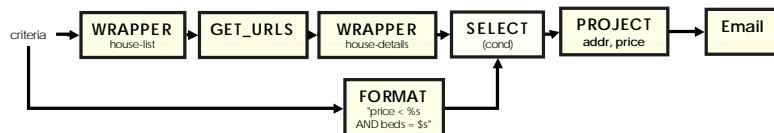
Craig Knoblock

University of Southern California

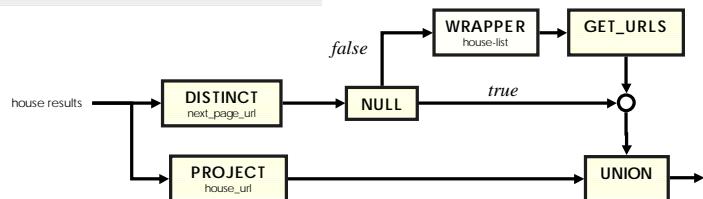
31

## Real Estate Plan

### FIND\_HOUSES



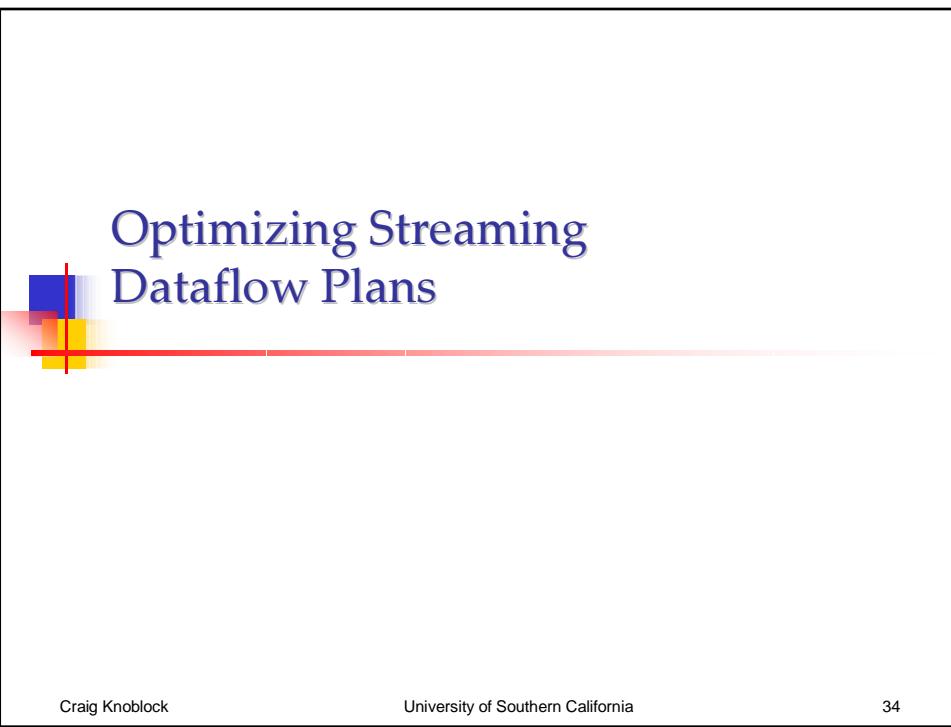
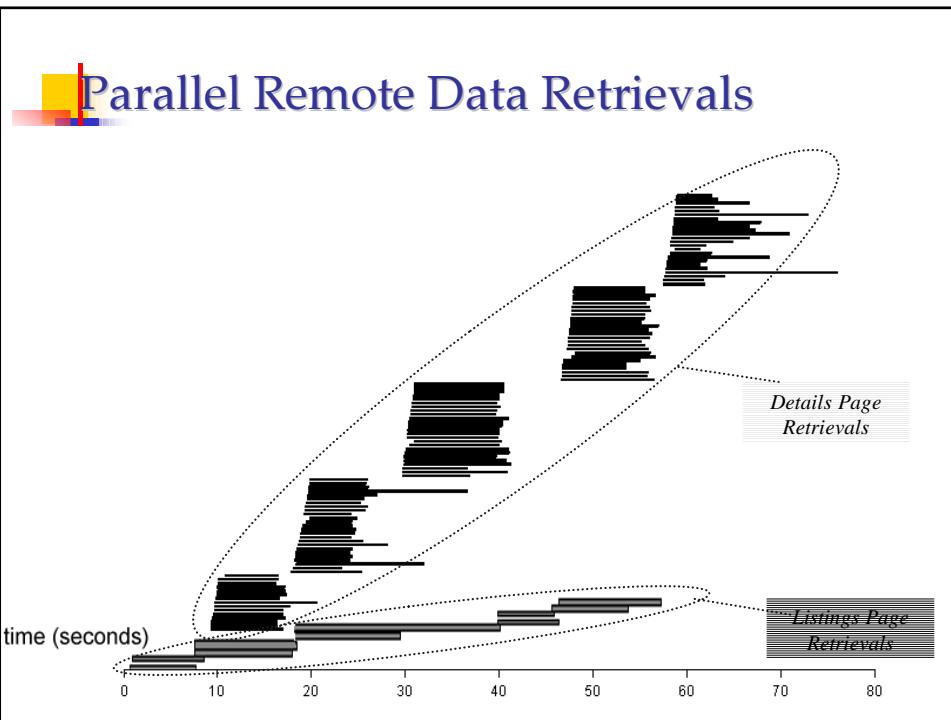
### GET\_URLS



Craig Knoblock

University of Southern California

32



## Adaptive Query Execution

- Network Query Engines
  - Tukwila (Ives et al., 1999)
    - Operator reordering
    - Optimized operators
  - Telegraph (Hellerstein et al. 2000)
    - Tuple-level adaptivity
  - Niagara (Naughton, DeWitt, et al. 2000)
    - Partial results for blocking operators
- Agent Execution Systems
  - Theseus (Barish & Knoblock, 2002)
    - Speculative execution

Craig Knoblock

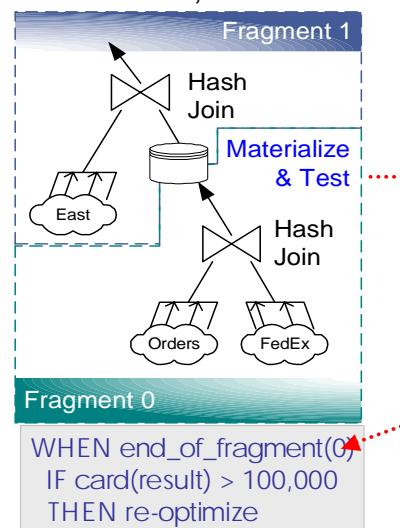
University of Southern California

35

## Interleaved Planning and Execution

From Ives et al., SIGMOD'99

- Generates initial plan
- Can generate partial plans and expand them later
- Uses rules to decide when to reoptimize



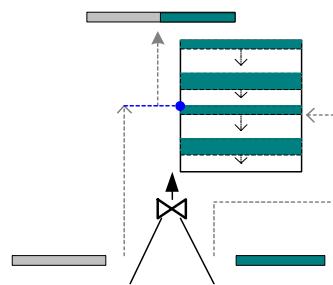
Craig Knoblock

University of Southern California

36

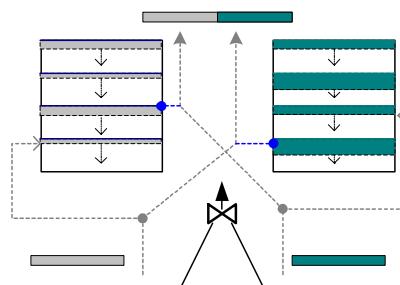
## Adaptive Double Pipelined Hash Join Operator

From Ives et al., SIGMOD'99



Hybrid Hash Join

- No output until inner read
- Asymmetric (inner vs. outer)



Double Pipelined Hash Join

- Outputs data immediately
- Symmetric
- More memory

Craig Knoblock

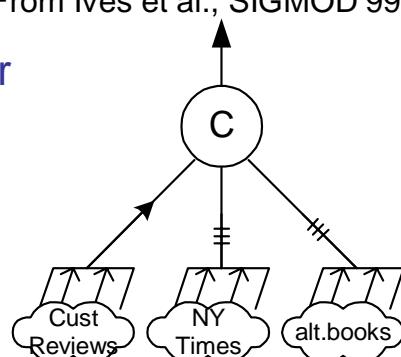
University of Southern California

37

## Dynamic Collector Operator

From Ives et al., SIGMOD'99

- Smart union operator
- Supports
  - Timeouts
  - slow sources
  - overlapping sources



```
WHEN  
timeout(CustReviews)  
DO activate(NYTimes),  
activate(alt.books)
```

Craig Knoblock

University of Southern California

38

## Tuple-level Adaptivity

(Hellerstein et al. 2000)

- Optimize horizontal parallelism
  - Adaptive dataflow on clusters (ie, data partitioning)
- Optimize vertical parallelism
  - Leverage commutative property of query operators to dynamically route tuples for processing
  - Result: adaptive streaming

Craig Knoblock

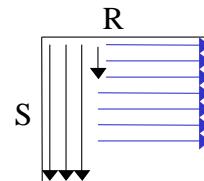
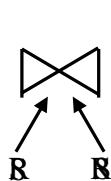
University of Southern California

39

## When can processing order be changed?

- Moment of symmetry:
  - Inputs can be swapped without state management
  - Nested Loops: at the end of each inner loop
  - Merge Join: any time
  - Hybrid Hash Join: never!

From Avnur & Hellerstein,  
SIGMOD 2000



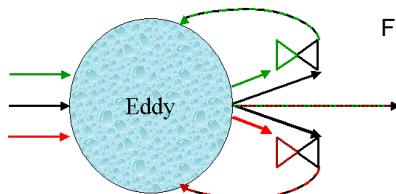
Craig Knoblock

University of Southern California

40

## Beyond Reordering Joins

Eddy



From Avnur & Hellerstein,  
SIGMOD 2000

- A pipelining tuple-routing iterator (just like join or sort)
- **Adjusts flow adaptively**
  - Tuples flow in different orders
  - Visit each op once before output
- **Naïve routing policy:**
  - All ops fetch from eddy as fast as possible
  - Previously-seen tuples precede new tuples

Craig Knoblock

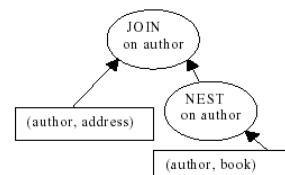
University of Southern California

41

## Execution with partial results

[Shanmugasundaram et al. 2000]

- Query execution involves evaluation of partial results
  - Reduces blocking nature of aggregation or joins
- **Basic idea**
  - Execute future operators as data streams in, refine as slow operators catch up
  - Execution is still driven by **availability of real data**
  - Notion of refinement is similar to "correction" in speculative execution



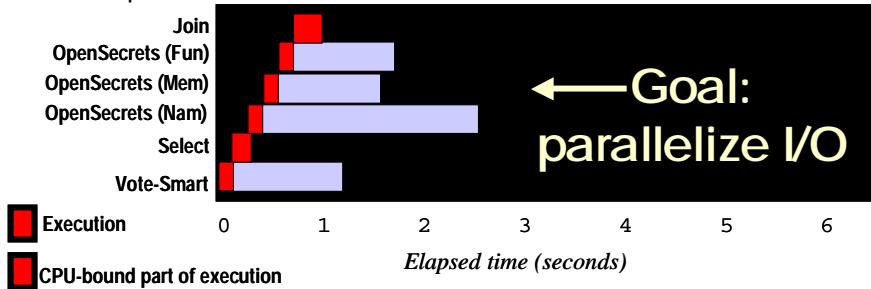
Craig Knoblock

University of Southern California

42

## Speculative Execution

- Standard streaming dataflow execution
  - Still I/O-bound (most operators are I/O-bound), CPU underused
  - Binding patterns compound delays
- To further increase parallelism: speculate about execution
  - Use earlier data as hints to speculatively execute downstream operators



Craig Knoblock

University of Southern California

43

## Speculating about plan execution

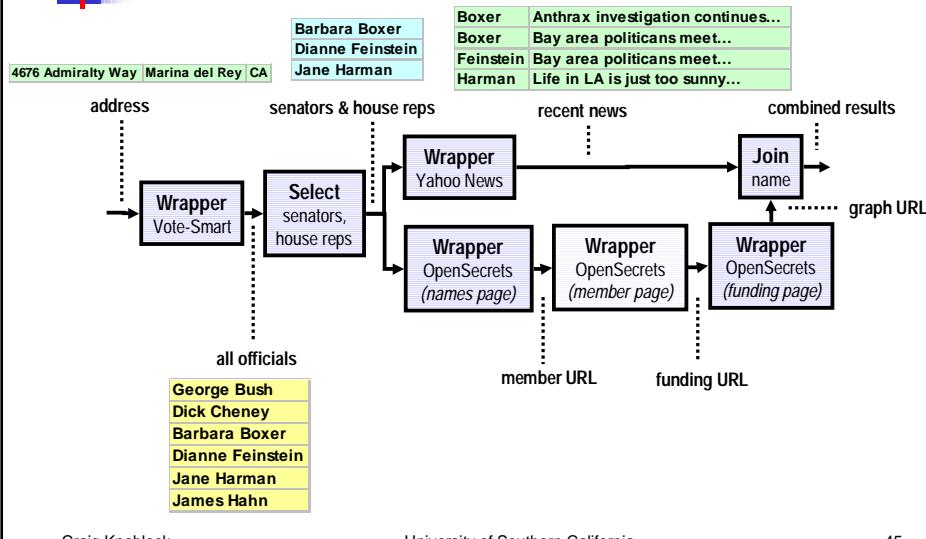
- Speculate about input to plan operators
  - Increase the level of operator-level parallelism
- Research questions
  - How to speculate?
    - What mechanism allows speculation to occur?
  - When to speculate?
    - What triggers speculation?
  - What to speculate about?
    - How do we predict data?
- Additional challenges
  - Maintaining correctness and fairness

Craig Knoblock

University of Southern California

44

## RepInfo agent plan



Craig Knoblock

University of Southern California

45

## Execution performance

- Measuring performance
  - Amdahl's law
    - Execution is only as fast as the costliest linear sequence
- Thus:
  - Slowest single data flow = fastest possible overall performance

Flow	Time
Vote-Smart, Select, Yahoo, Join	3.3 sec
Vote-Smart, Select, OpenSecrets, Join	6.2 sec

- Execution time = MAX (3.3, 6.2) = 6.2 sec

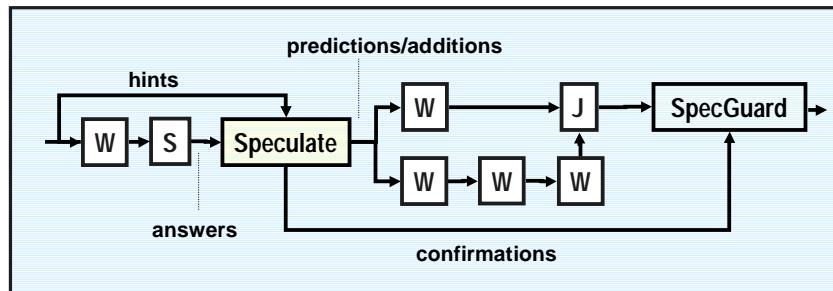
Craig Knoblock

University of Southern California

46

## Overview of approach

- Automatically augment plan with 2 operators
  - Speculate: Makes predictions and corrections
  - SpecGuard: Halts errant speculation



Craig Knoblock

University of Southern California

47

## Resulting performance

- ReplInfo (original plan)
  - Execution time: 6.2 sec
- ReplInfo-Spec
  - Individual flow performance:

Flow	Time
Vote-Smart, Select	1.4 sec
Yahoo, Join	1.9 sec
OpenSecrets, Join	4.8 sec

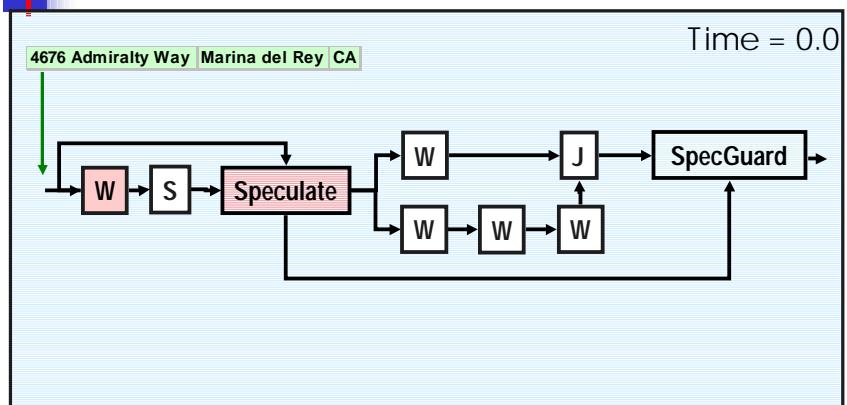
- Thus, execution time is now 4.8 sec
  - Speedup = ( 6.2 / 4.8 ) = 1.3

Craig Knoblock

University of Southern California

48

## Plan execution starts

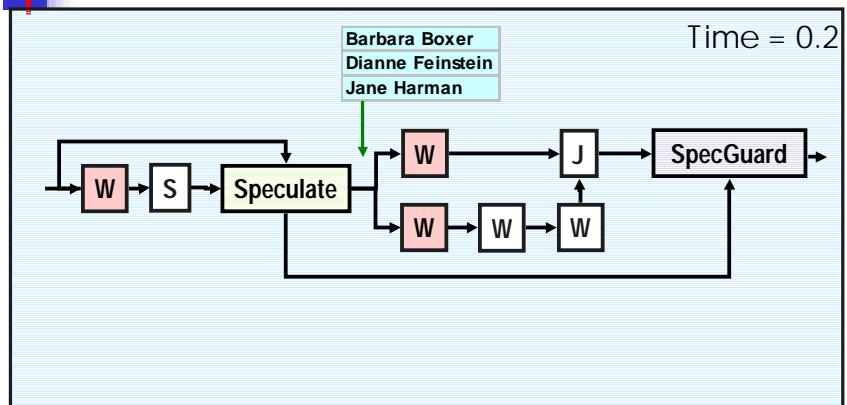


Craig Knoblock

University of Southern California

49

## Speculation about representatives

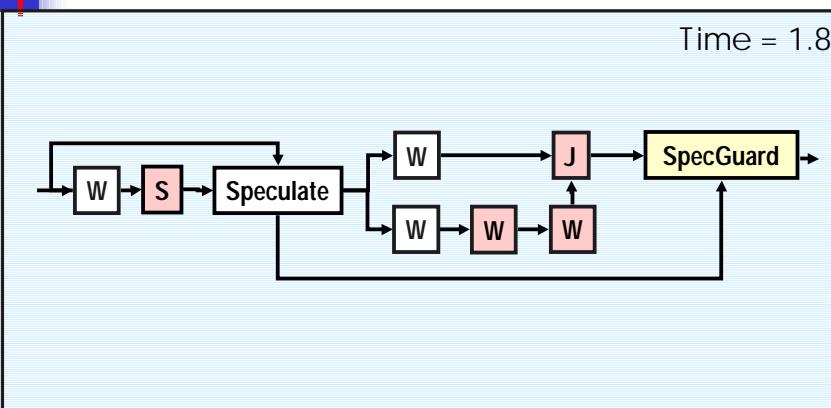


Craig Knoblock

University of Southern California

50

## Speculation results received

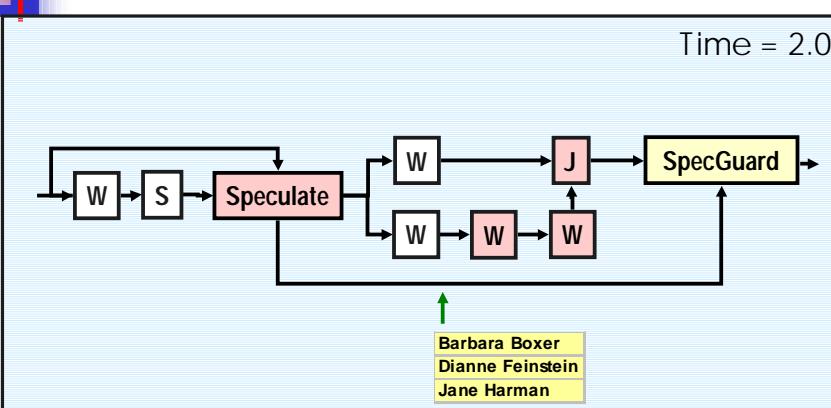


Craig Knoblock

University of Southern California

51

## Speculation results received

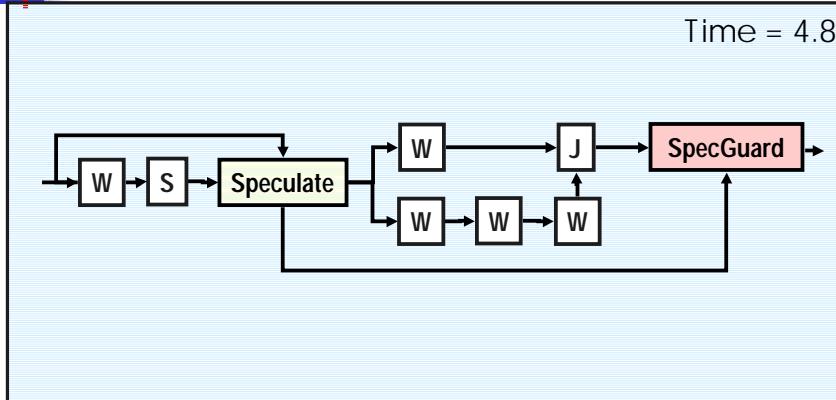


Craig Knoblock

University of Southern California

52

## Confirming speculation



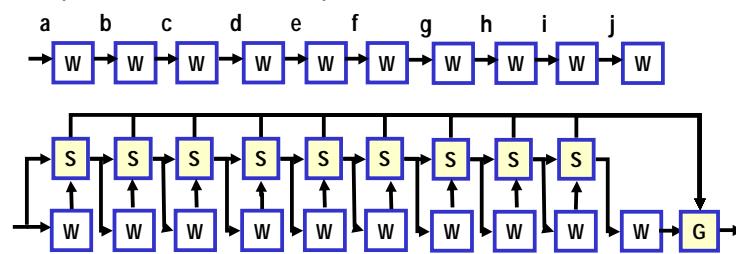
Craig Knoblock

University of Southern California

53

## Cascading speculation

- Major limitation thus far:
  - We are only speculating once
- Cascading speculation
  - Speculation based on speculation



- Theoretical speedup of above example=  $(10/1)= 10$

Craig Knoblock

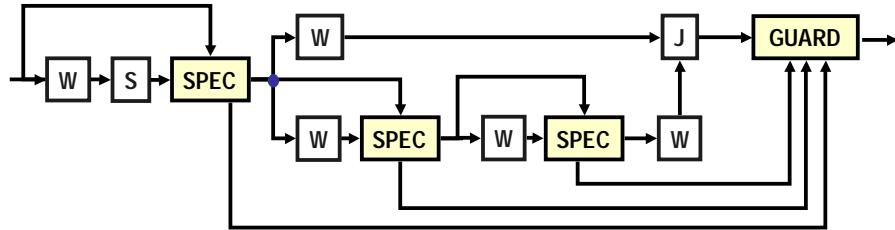
University of Southern California

54

## Cascading speculation

- ReplInfo Example:

- Use predicted officials to speculate about the OpenSecrets member and funding URLs



- Estimated performance

- Slowest existing flow =  $\text{MAX}(1.4, 1.9, 1.4, 2.4) = \underline{\text{2.4 seconds}}$
- Speedup =  $(6.2 / 2.4) = \text{2.59}$

Craig Knoblock

University of Southern California

55

## Ensuring correctness and fairness

### Correctness

- SpecGuard does this
- Never emits tuples unless confirmed
- Must be placed prior to
  - Plan exit
  - Any operators that change the external world

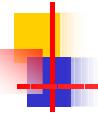
### Fairness

- Speculation must never usurp normal execution
- Plan execution involves multiple concurrent threads
  - Operators are associated with individual threads
- One simple solution:
  - Make Speculate and SpecGuard lower priority threads
  - Let the CPU handle fair scheduling

Craig Knoblock

University of Southern California

56



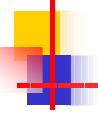
## Where and when to speculate?

- Generally speaking:
  - Speculate about those operators that are:
    - Dynamic (not FDs)
    - Not the initial set of operators executed
- Remember: Dataflow  $\neq$  von-Neumann
  - Execution is not sequential
  - Instead: a set of independent data flow paths
- Amdahl's law
  - Most expensive path (MEP) is the prime concern
  - Optimizing anything BUT the MEP is a waste

Craig Knoblock

University of Southern California

57



## Automatic plan augmentation

- Focus on most expensive path (MEP)
  - Specifically on bottleneck operators (e.g., Wrapper)
- Algorithm sketch
  - Locate MEP
  - Find "best" candidate transformation for that path
  - If no candidate found, then exit
  - Transform plan accordingly
  - Repeat
- Finding the "best" candidate
  - Identify path with highest likely average execution time

Craig Knoblock

University of Southern California

58

## The challenge

- We need to be able to predict data
- Example
  - Predict federal officials given an address
- Categories of predictions

Category	Hint	Prediction
A	Previously seen	Previously seen
B	Never seen	Previously seen
C	Never seen	Never seen
- How do we deal with...?
  - Prediction given new hints
  - Making new predictions

Craig Knoblock

University of Southern California

59

## Caching

- Associate answers with previously seen hints

Key	Value
4676 Admiralty Way, Marina del Rey, CA, 90292	Boxer, Feinstein, Harman
14044 Panay Way, Marina del Rey, CA 90292	Boxer, Feinstein, Harman
4065 Lincoln Blvd, Venice, CA 90405	Boxer, Feinstein, Waxman
- Method of prediction
  1. When hint arrives, locate value in table
  2. If hint not in table, do not issue prediction
  3. Otherwise, predict the value found
- Problems
  - Only handles predictions of category A
    - Cannot deal with new hints or issue new predictions
  - Space inefficient

Craig Knoblock

University of Southern California

60

## Decision trees

- Can be used to learn that, when predicting officials,  
→ city and zip are key attributes

hint

answer

Street	City	State	Zip	Representative
14044 Panay Way	Marina del Rey	CA	90292	Jane Harman
4676 Admiralty Way	Marina del Rey	CA	90292	Jane Harman
101 Washington Blvd	Venice	CA	90292	Jane Harman
1301 Main St	Venice	CA	90291	Jane Harman
1906 Lincoln Blvd	Venice	CA	90291	Jane Harman
2107 Lincoln Blvd	Santa Monica	CA	90405	Henry Waxman
2222 S Centinela Ave	Los Angeles	CA	90064	Henry Waxman
4065 Glencoe Ave	Marina del Rey	CA	90292	Diane Watson
3970 Berryman Ave	Los Angeles	CA	90066	Diane Watson
11461 Washington Blvd	Los Angeles	CA	90066	Diane Watson

city = Marina del Rey: Jane Harman (2)  
city = Venice: Jane Harman (3)  
city = Santa Monica: Henry Waxman (1)  
city = Los Angeles:  
....zip <= 90064: Henry Waxman (1)  
zip > 90064: Diane Watson (2)

- Since prediction is based on subset of attributes  
→ prediction given new hints is possible

Craig Knoblock

University of Southern California

61

## Transducers for hint translation

- Recall that we want to be able to predict
  - <http://www.opensecrets.org/politicians/summary.asp?CID=N00007364>
  - <http://www.opensecrets.org/politicians/sector.asp?CID=N00007364>
- Prediction viewed as a translation
  - Simple subsequential transducers are used in NLP research for language translation
  - General idea
    - Construct alignment between tokens of L1 and L2
    - Build transducers that generate L2 sentences from L1 sentences
      - Transduction can be applied at the word or letter level

Craig Knoblock

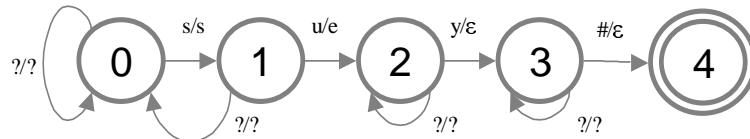
University of Southern California

62

## Transducers for hint translation

- Example
    - Construct alignment

- Build transducer



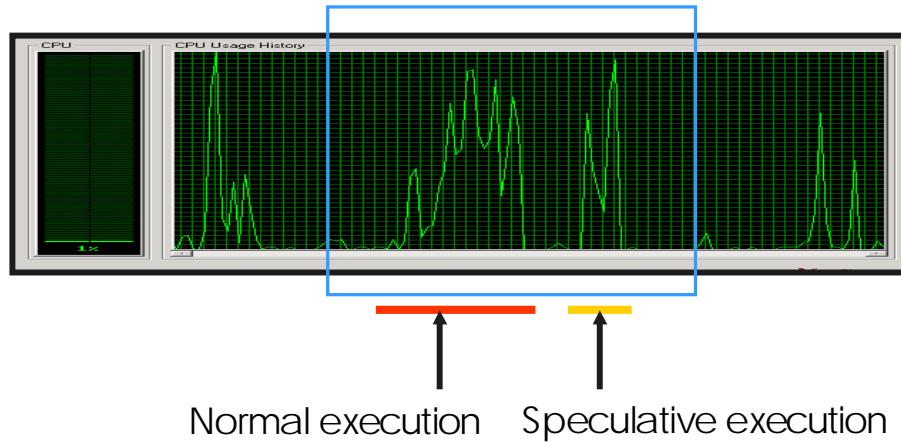
Craig Knoblock

University of Southern California

63

## Experimental results

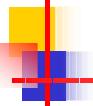
- CPU impact of sample run



Craig Knoblock

University of Southern California

64



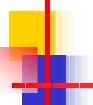
## Discussion

- Theseus, Tukwila, Telegraph, Niagara are all:
  - Streaming dataflow systems
  - Target network-based query execution
    - Large source latencies
    - Unknown characteristics of sources
  - Focus on techniques for improving the efficiency of plan execution
- Challenges in Plan Execution
  - How to interleave planning and execution
  - How to interleave sensing actions
  - Other approaches to improve performance
  - Improved techniques for making predictions

Craig Knoblock

University of Southern California

65



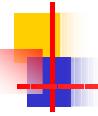
## Bibliography

- Dataflow computing
  - Foundations
    - Dennis, Jack B. (1974). First version of a data-flow procedure language. *Lecture Notes in Computer Science* vol. 19, pp 362–376.
    - Arvind and R.S. Nikhil (1990). Executing a program on the MIT tagged-token dataflow architecture. *IEEE Transactions on Computers* (1990), pp 300–318.
  - Dataflow / von Neumann hybridization
    - Iannucci, Robert A. (1988) Toward a dataflow/von Neumann hybrid architecture. In *Proceedings of the 19th Annual International Conference on Computer Architecture (ICSA)*, pp 131–140.
    - Papadopolous, Gregory M. and Kenneth R. Traub. (1991) Multithreading: a revisionist view of dataflow architectures. In *Proceedings of the 18th Annual Symposium on Computer Architecture*, pp 342–351.

Craig Knoblock

University of Southern California

66



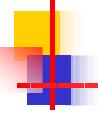
## Bibliography

- Parallel database systems
  - Shared nothing architectures
    - DeWitt, David J. and Jim Gray (1992). Parallel database systems: the future of high-performance database systems. Communications of the ACM 35(6), pp 85-98.
  - Parallel query execution
    - Wilschut, Annita N. and Peter M.G. Apers. (1991) Dataflow query execution in a main memory environment. In Proceedings of the First International Conference on Parallel and Distributed Information Systems, pp 68–77.
    - Graefe, Goetz (1994) Volcano – an extensible and parallel query evaluation system. IEEE Transactions on Knowledge and Data Engineering 6(1), pp 120–135 .

Craig Knoblock

University of Southern California

67



## Bibliography

- Network information gathering
  - Niagara
    - Naughton, Jeffrey F., David J. DeWitt, David Maier, and many others. (2001). The niagara internet query system. IEEE Data Engineering Bulletin 24(2): 27–33.
  - Telegraph
    - Hellerstein, Joseph M., Michael J. Franklin, Sirish Chandrasekaran, Amol Deshpande, Kris Hildrum, Sam Madden, Vijayshankar Raman and Mehul A. Shah (2000). Adaptive query processing: technology in evolution. IEEE Data Engineering Bulletin 23(2): 7--18.

Craig Knoblock

University of Southern California

68

## Bibliography

- Network information gathering
  - Theseus
    - Barish, Greg and Craig A. Knoblock. An expressive and efficient language for information gathering on the web. (2002) Proceedings of the Sixth International Conference on AI Planning and Scheduling Workshop: Is There Life Beyond Operator Sequencing? - Exploring Real-World Planning. pp. 5–12.
  - Tukwila
    - Ives, Zachary G., Daniela Florescu, Marc Friedman, Alon Levy and Daniel S. Weld (1999). An adaptive query execution system for data integration. In Proceedings of the ACM SIGMOD International Conference on Management of Data. pp 299–310.

Craig Knoblock

University of Southern California

69

## Bibliography

- Adaptive query processing
  - Adaptive tuple routing
    - Avnur, Ron and Joseph M. Hellerstein (2000). Eddies: continuously adaptive query processing. Proceedings of the ACM SIGMOD International Conference on the Management of Data. pp. 261--272.
  - Evaluation of partial results
    - Shanmugasundaram, Jayavel, Kristin Tufte, David J. DeWitt, Jeffrey F. Naughton and David Maier (2000). Architecting a network query engine for producing partial results. Proceedings of the ACM SIGMOD 3rd International Workshop on Web and Databases (WebDB). pp. 17-22.
    - Raman, Vijayshankar and Joseph M. Hellerstein (2002). Partial results for online query processing. Proceedings of the ACM SIGMOD International Conference on the Management of Data.
  - Speculative execution
    - Barish, Greg and Craig A. Knoblock (2002) Speculative execution for information gathering plans. In Proceedings of the Sixth International Conference on AI Planning and Scheduling, pp 259–268.

Craig Knoblock

University of Southern California

70