Fluxo

Improving the Responsiveness of Internet Services with Automatic Cache Placement

Alexander Rasmussen – UCSD (Presenting)
Emre Kiciman – MSR Redmond
Benjamin Livshits – MSR Redmond
Madanlal Musuvathi – MSR Redmond

UCSDCSE
Computer Science and Engineering

Microsoft Research

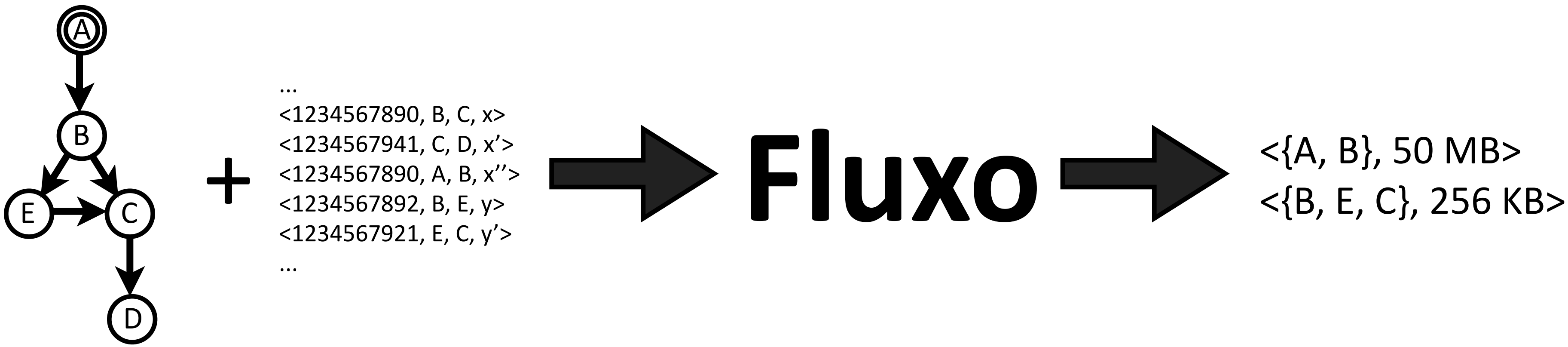
Caching in Internet Services

- Satisfying user request involves calling many external components, aggregating data
- Want to cache computation performed by some components to improve performance
  - Disk-intensive operations, DB queries, etc.
- What you cache and when depends on a number of factors
  - Workload, architecture, SLAs, ...

Caching in Internet Services

- Choice of what, where, how much to cache is usually very ad-hoc
  - Programmer intuition
  - Localized profiling
- "Best" choice can change rapidly over time; too quickly for humans to respond manually
- Need an automatic solution!

Fluxo - Automatic Cache Optimization

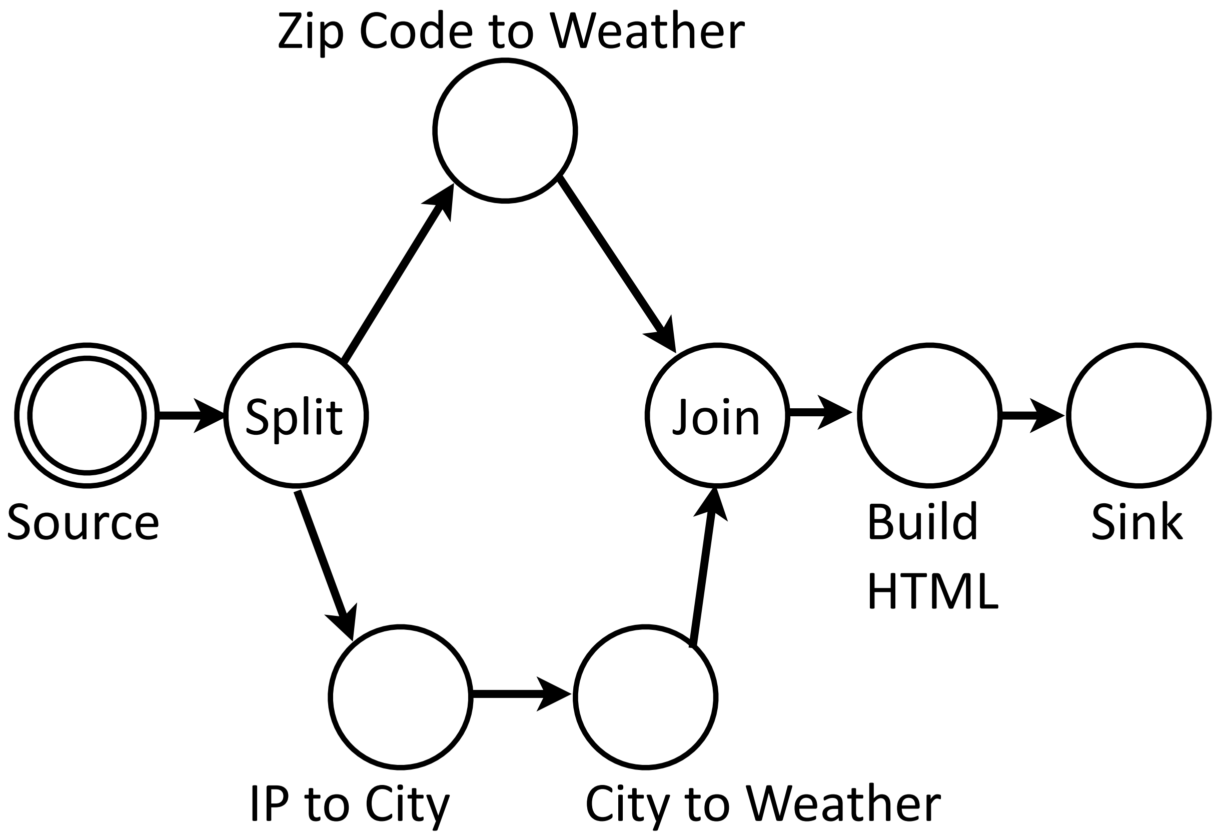


- Describe Internet service as dataflow graph
- Gather runtime request traces
- Simulate and optimize to converge on reasonably good cache placement policy

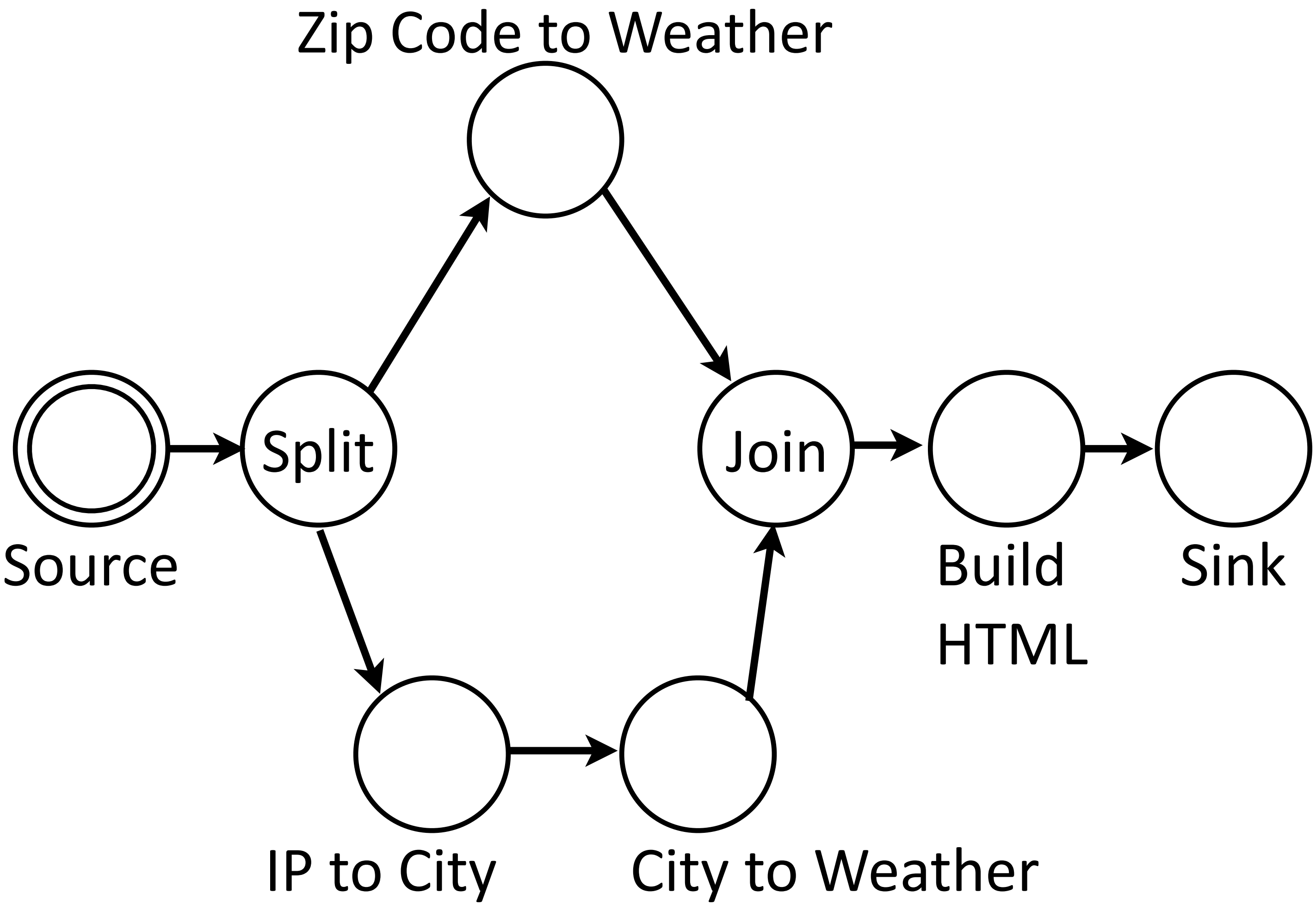
Fluxo Dataflow Graphs

- *Source* node produces request as tuple
- *Sink* node consumes response as tuple
- All other nodes are *components* which may call external services

Sample Service - Weather Report

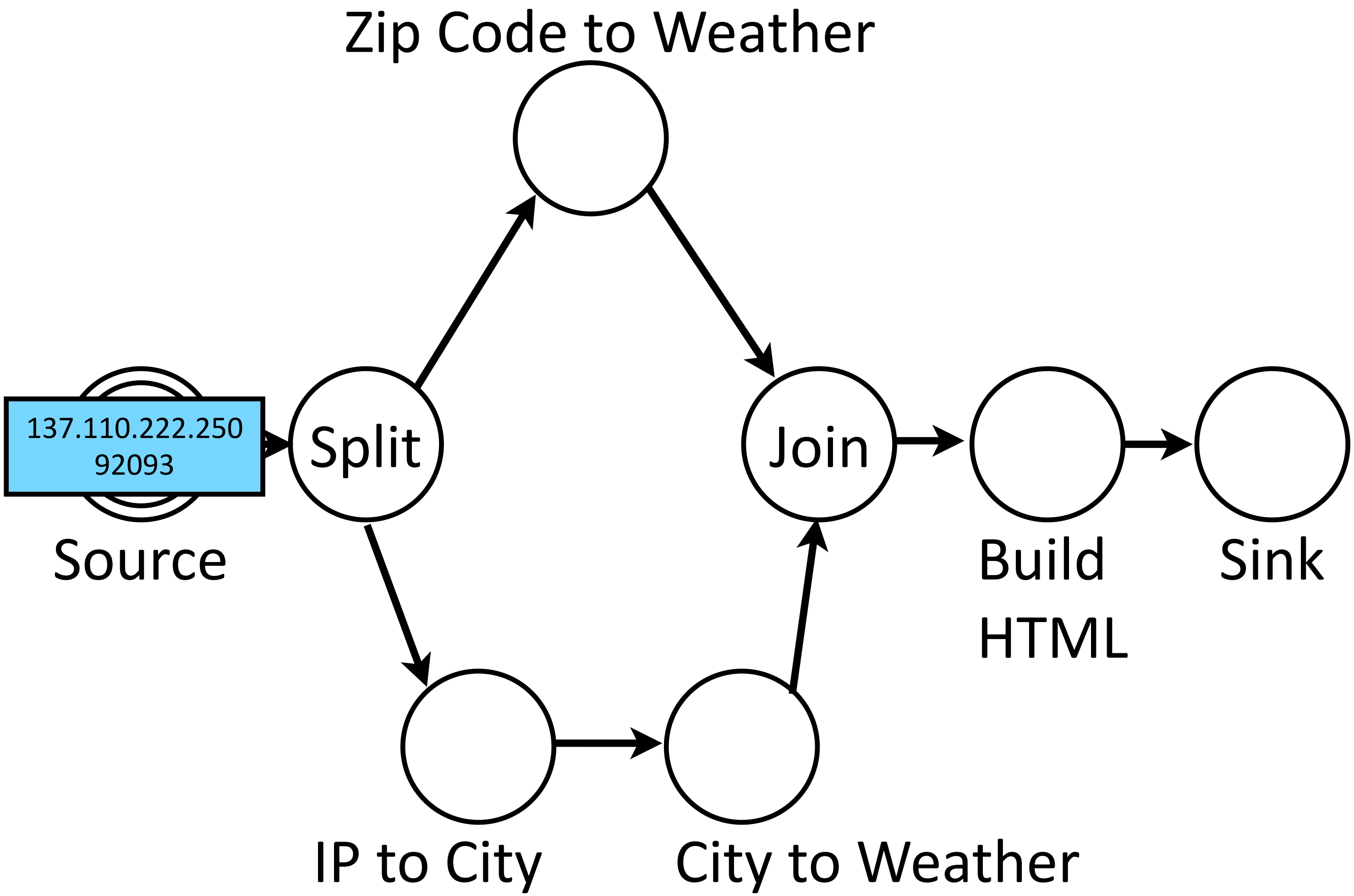


Weather Service - Sample Input

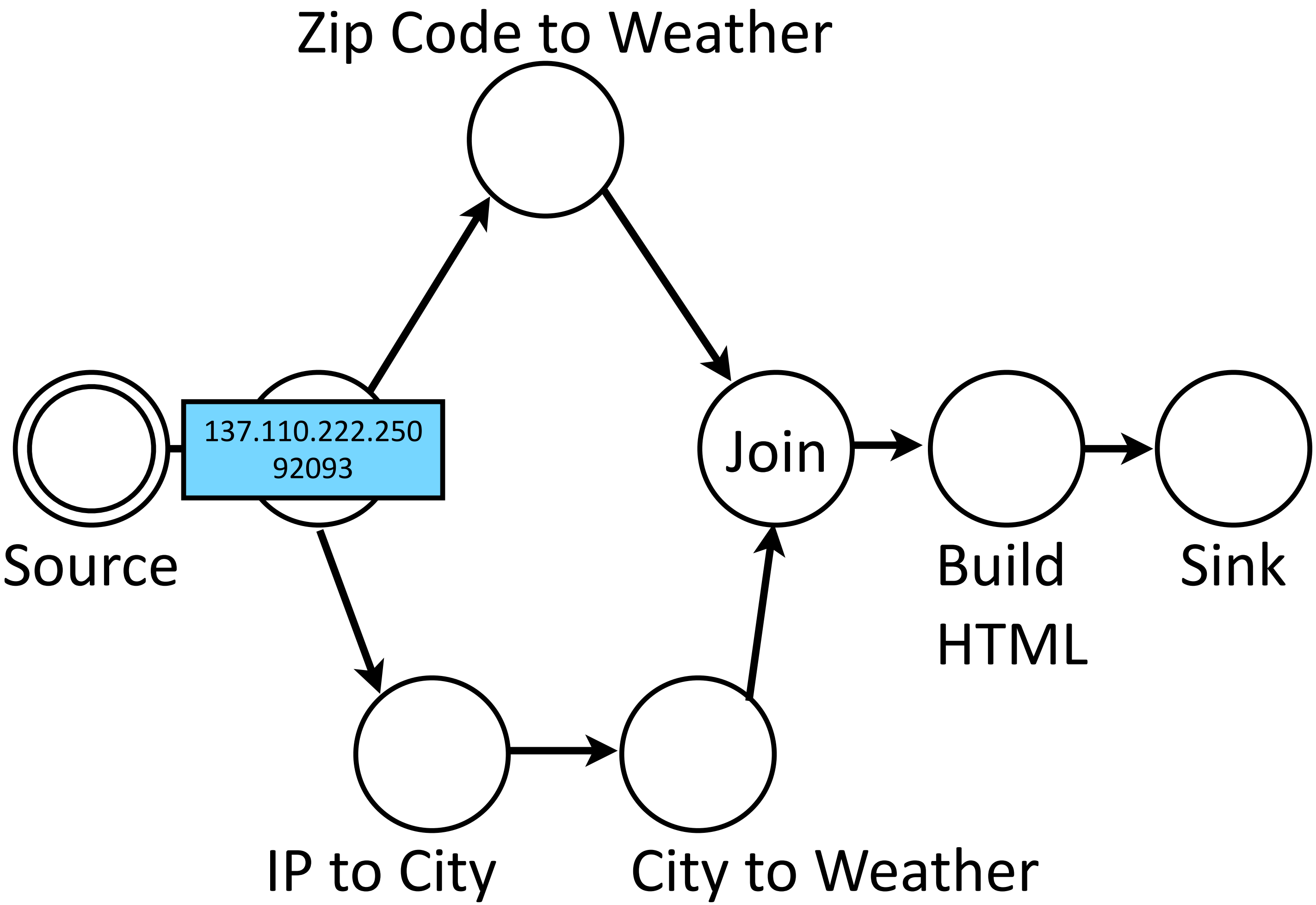


7

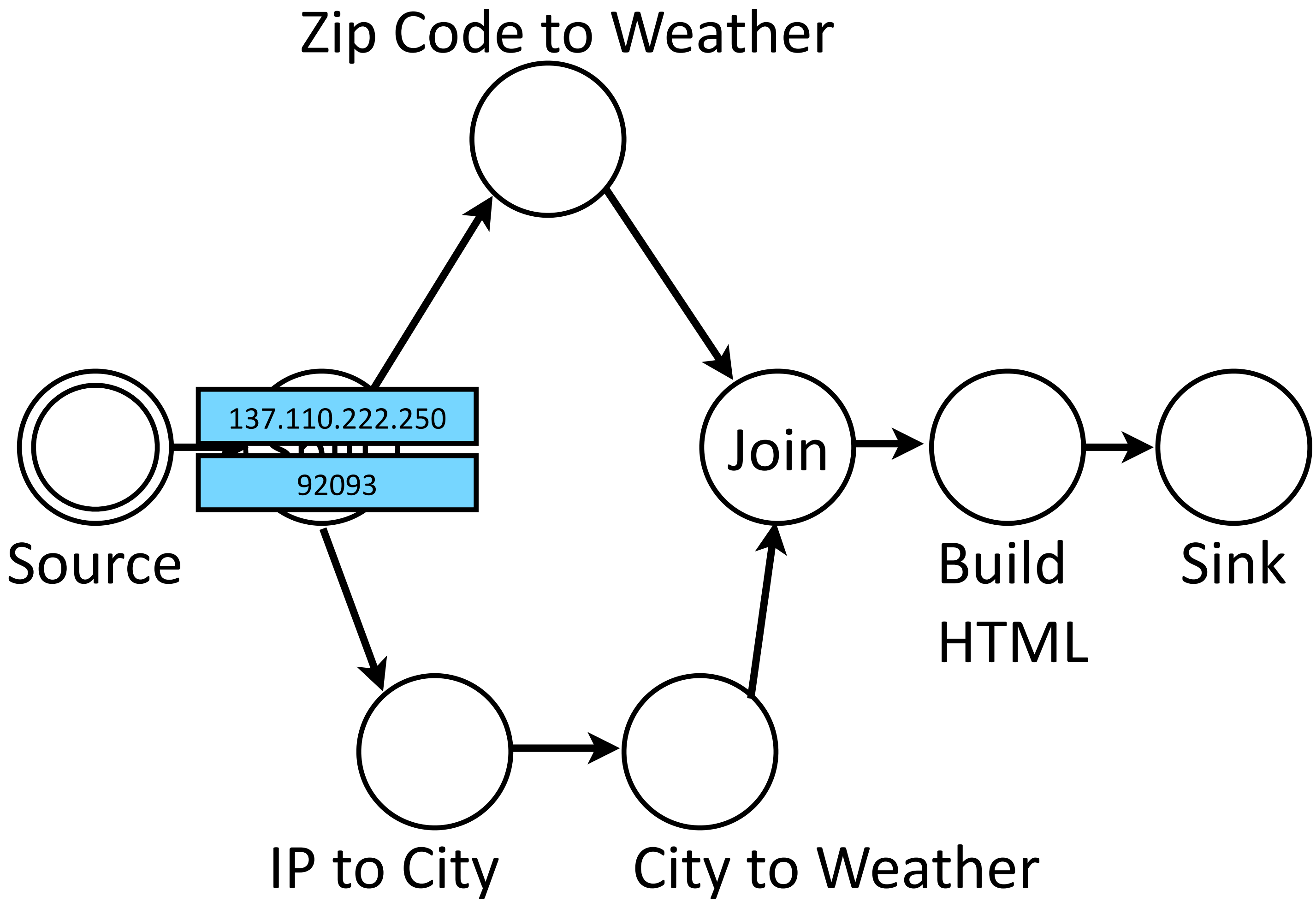
Weather Service - Sample Input



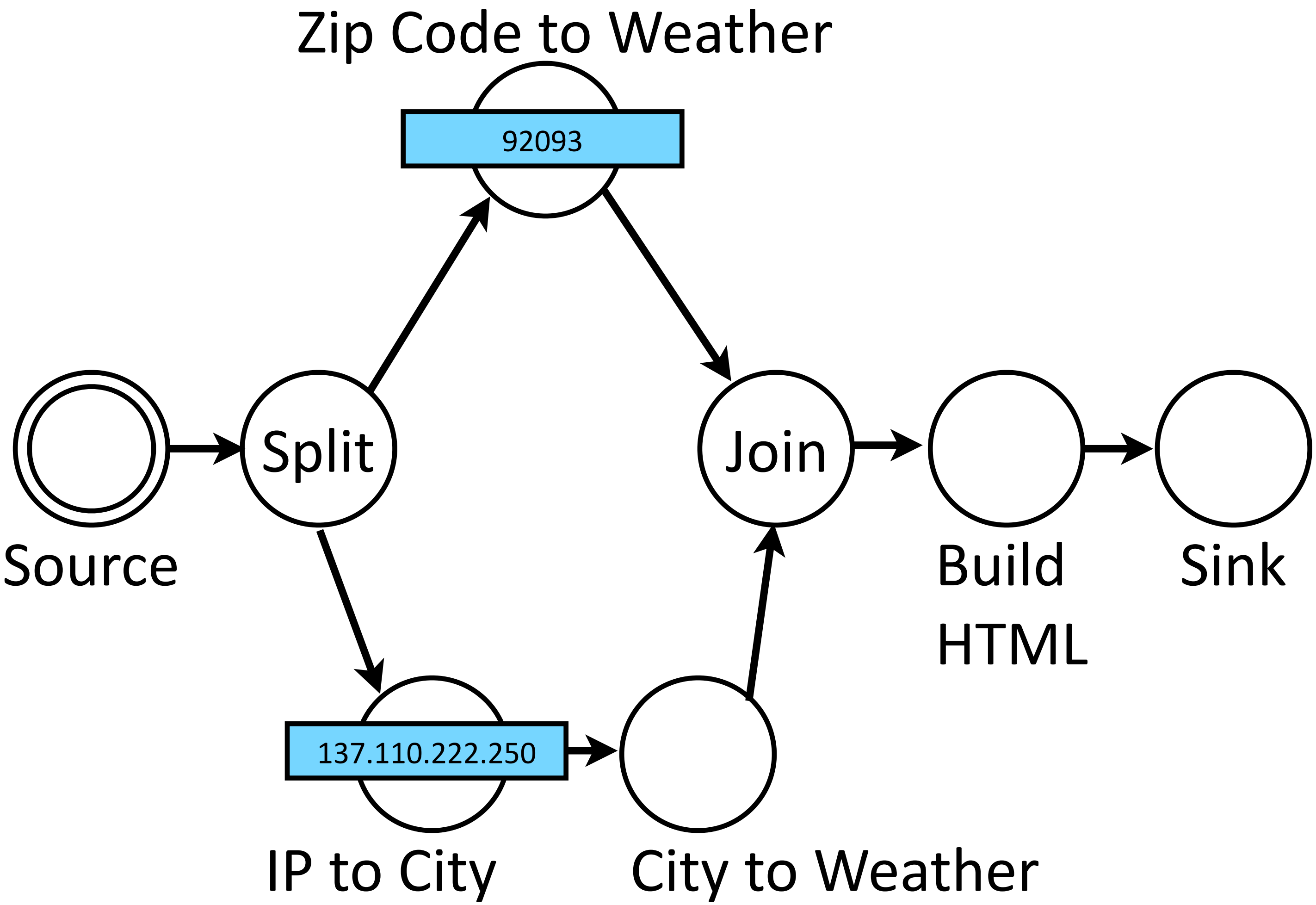
Weather Service - Sample Input



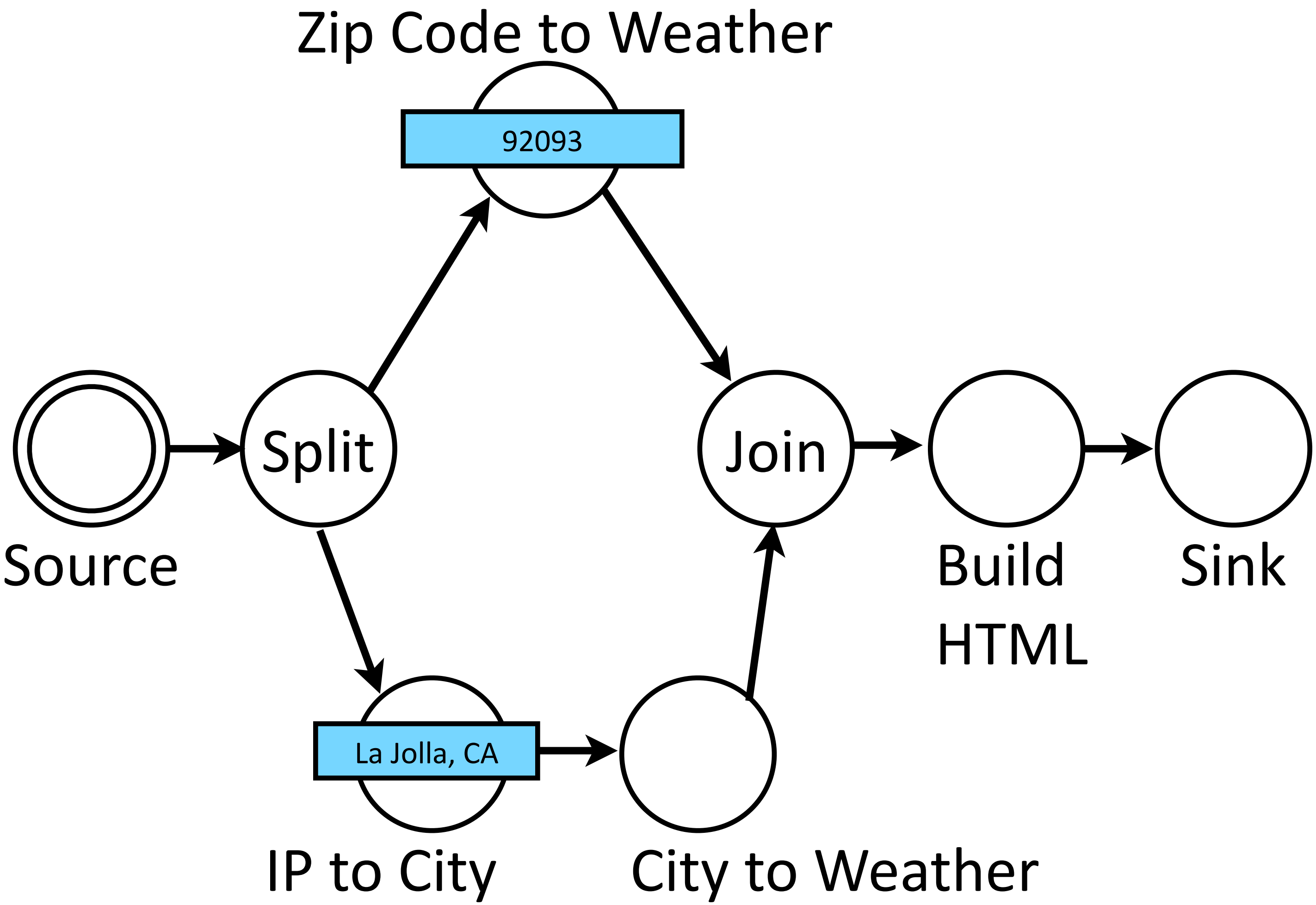
Weather Service - Sample Input



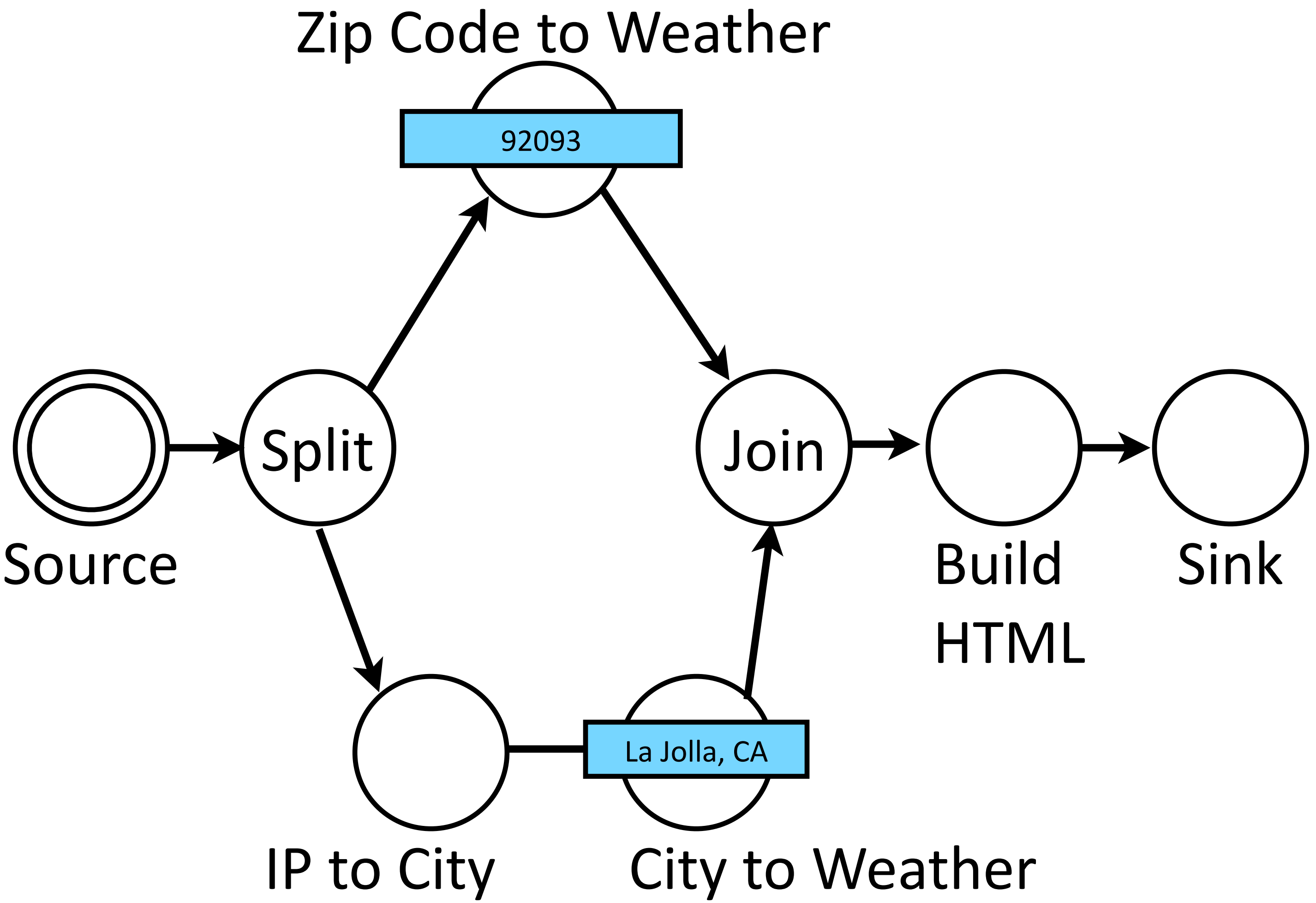
Weather Service - Sample Input



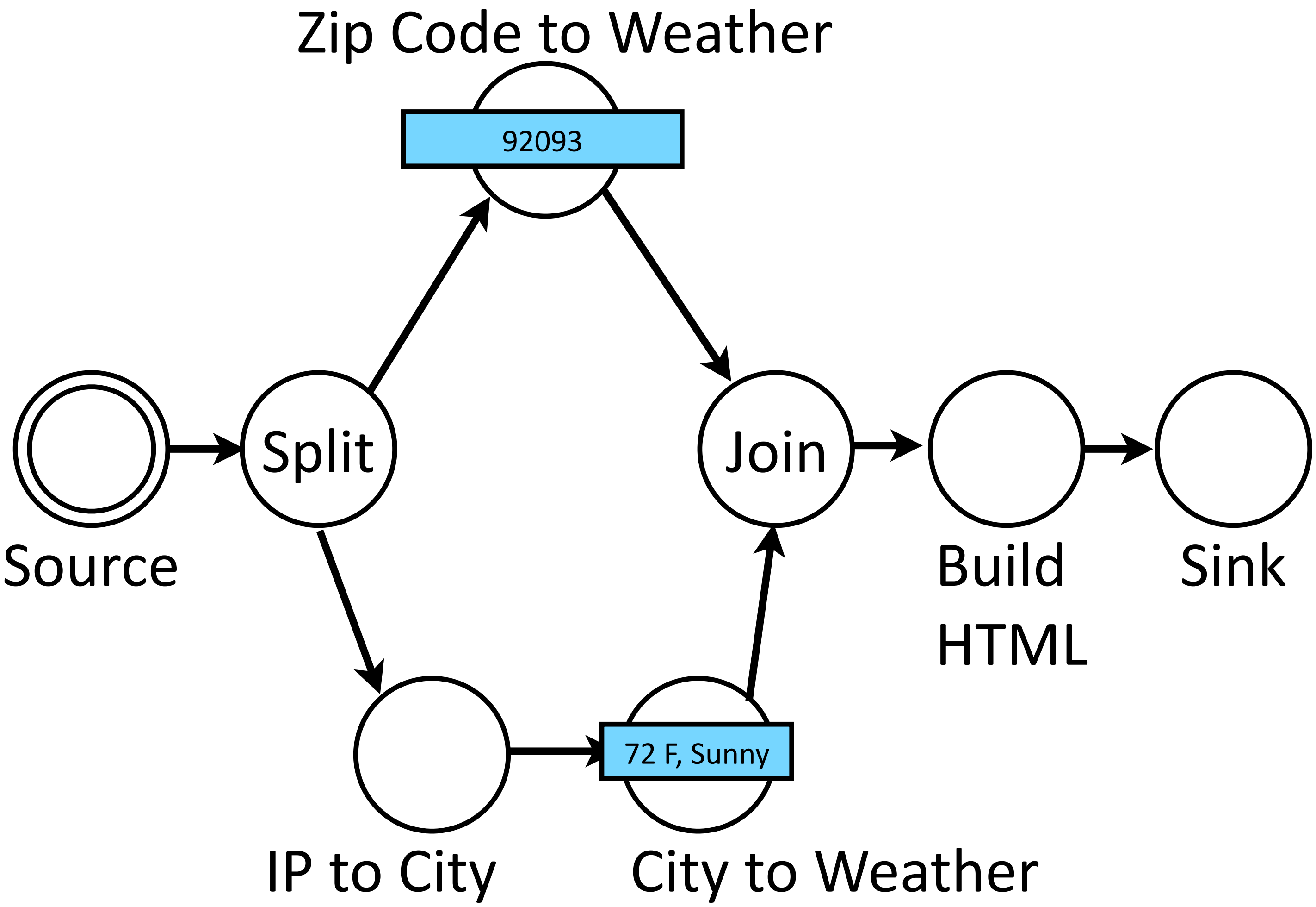
Weather Service - Sample Input



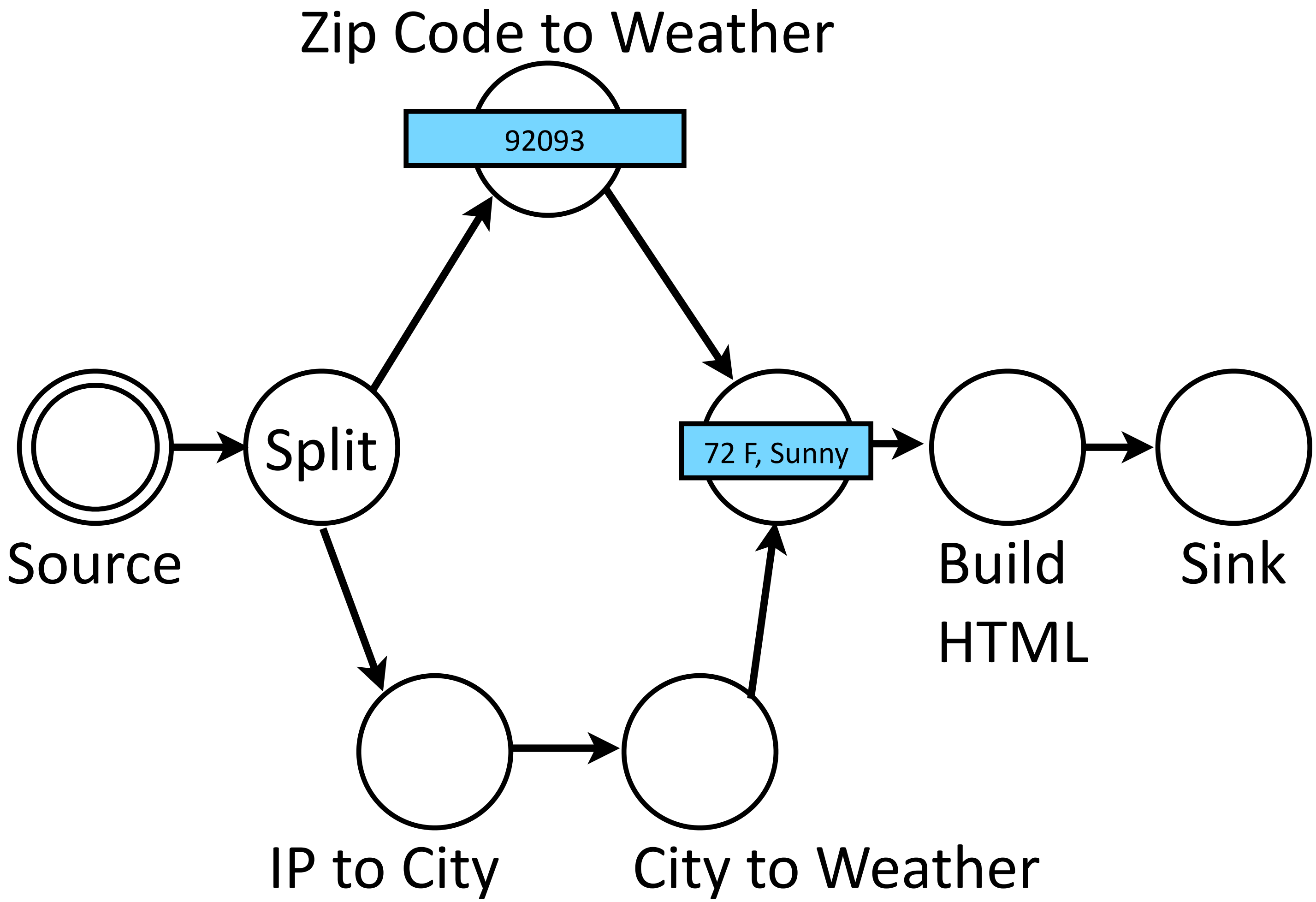
Weather Service - Sample Input



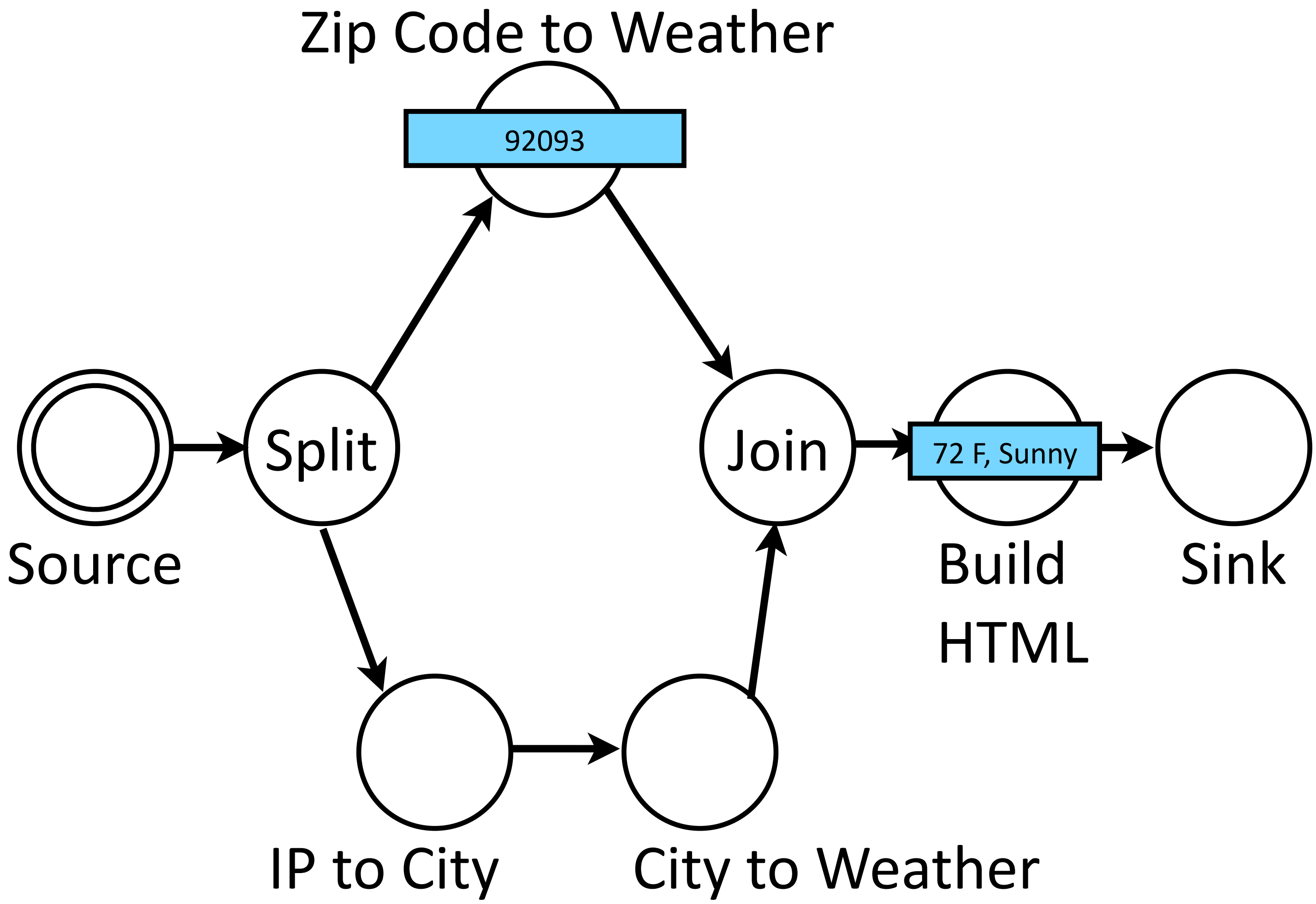
Weather Service - Sample Input



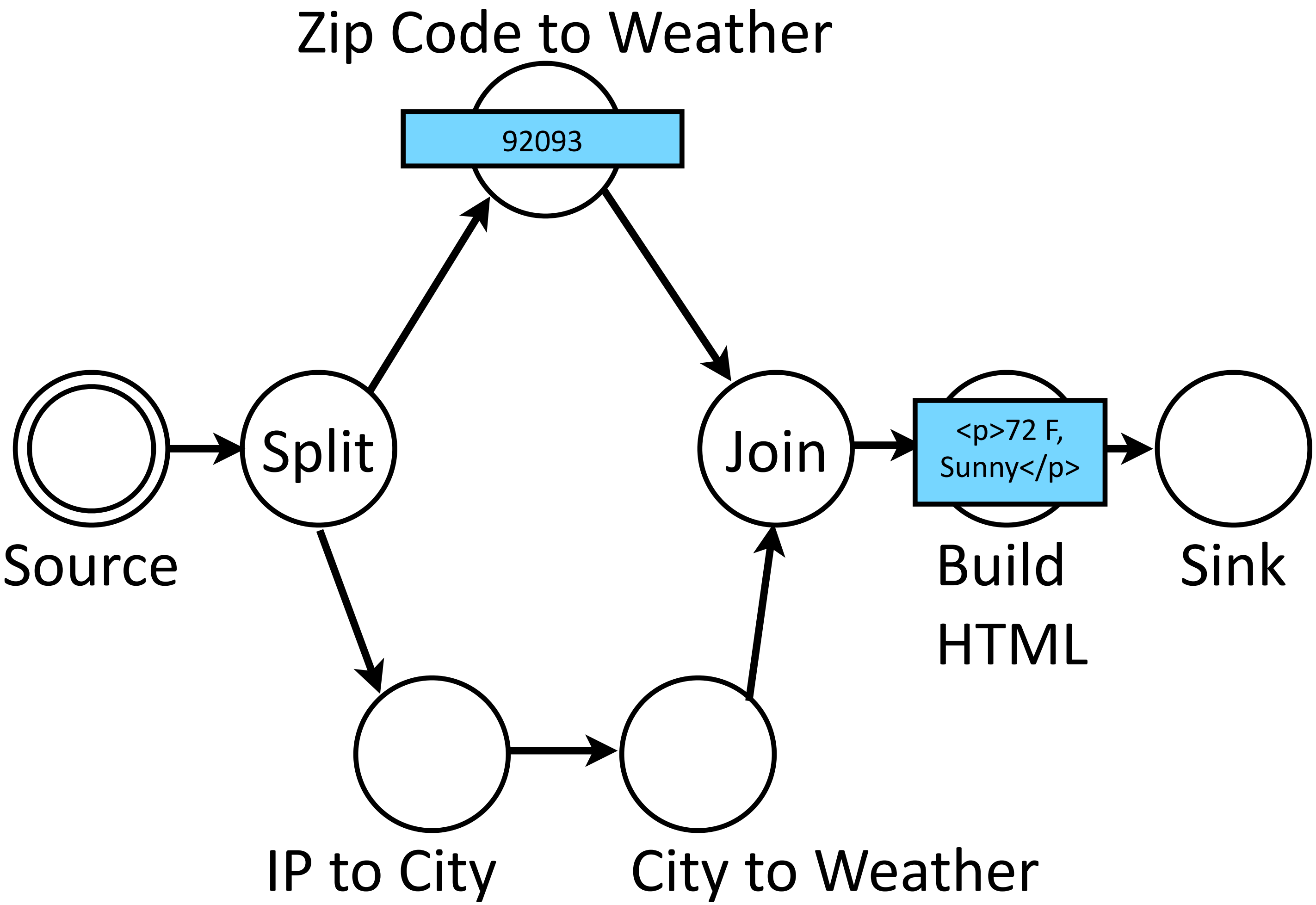
Weather Service - Sample Input



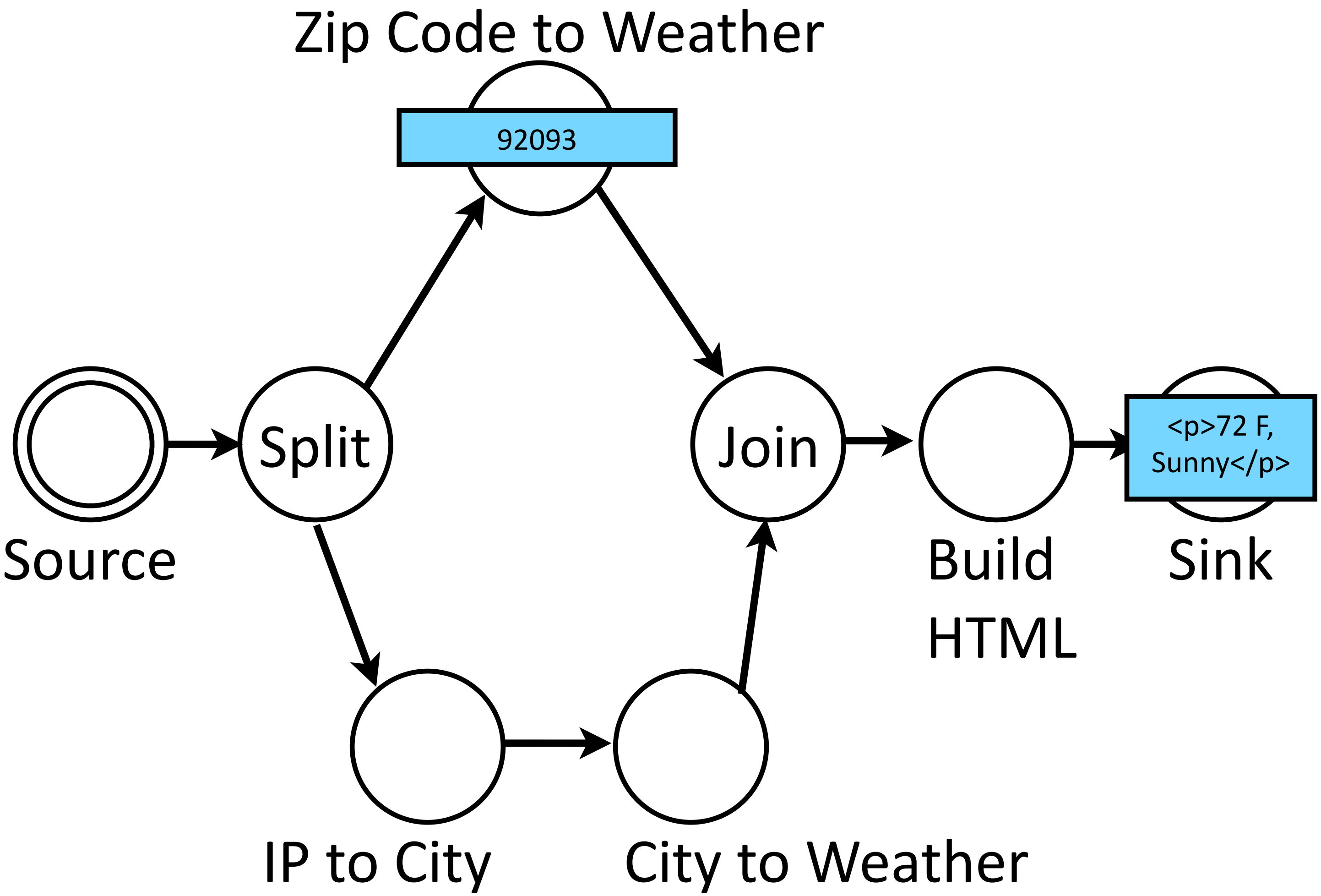
Weather Service - Sample Input



Weather Service - Sample Input

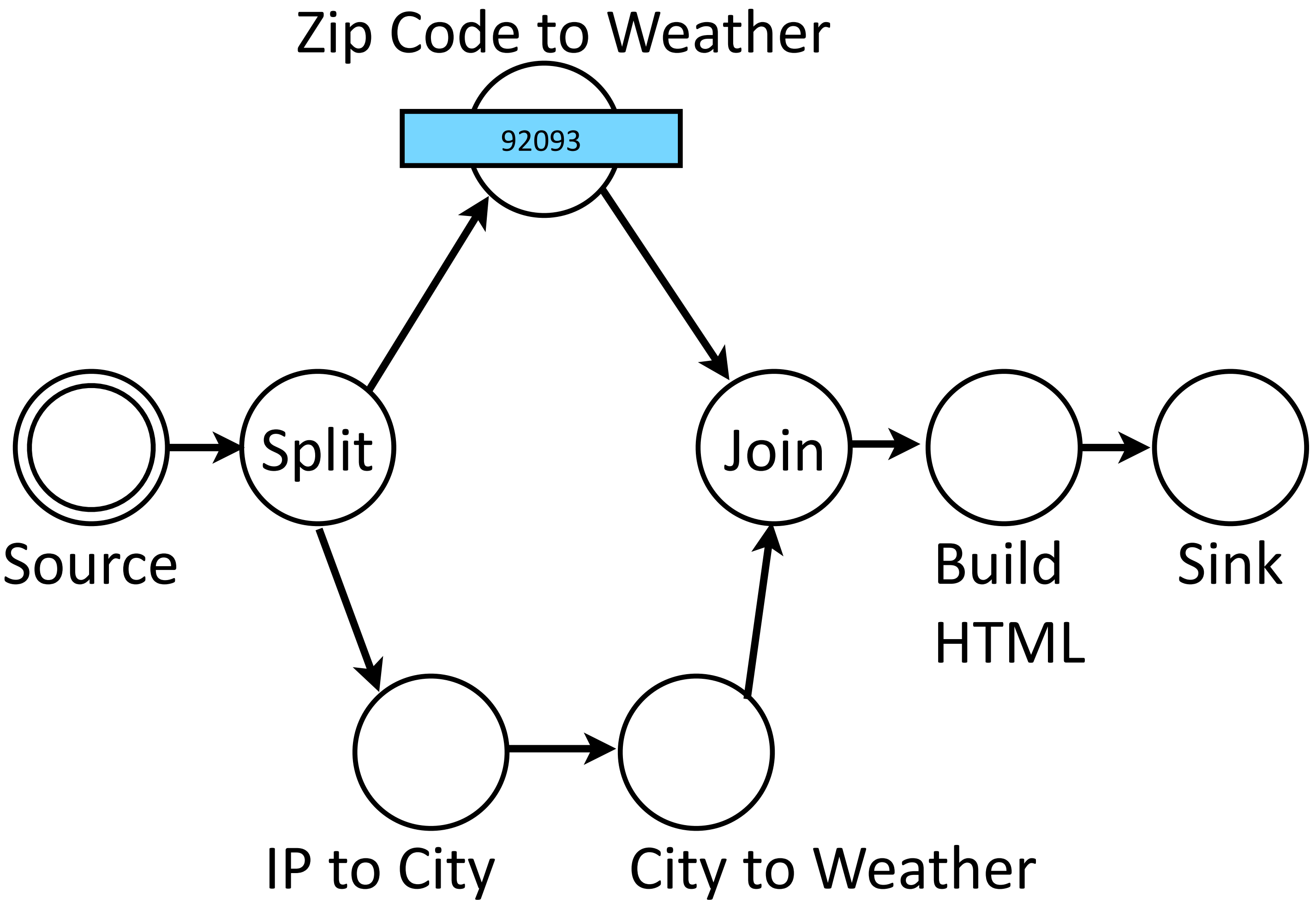


Weather Service - Sample Input



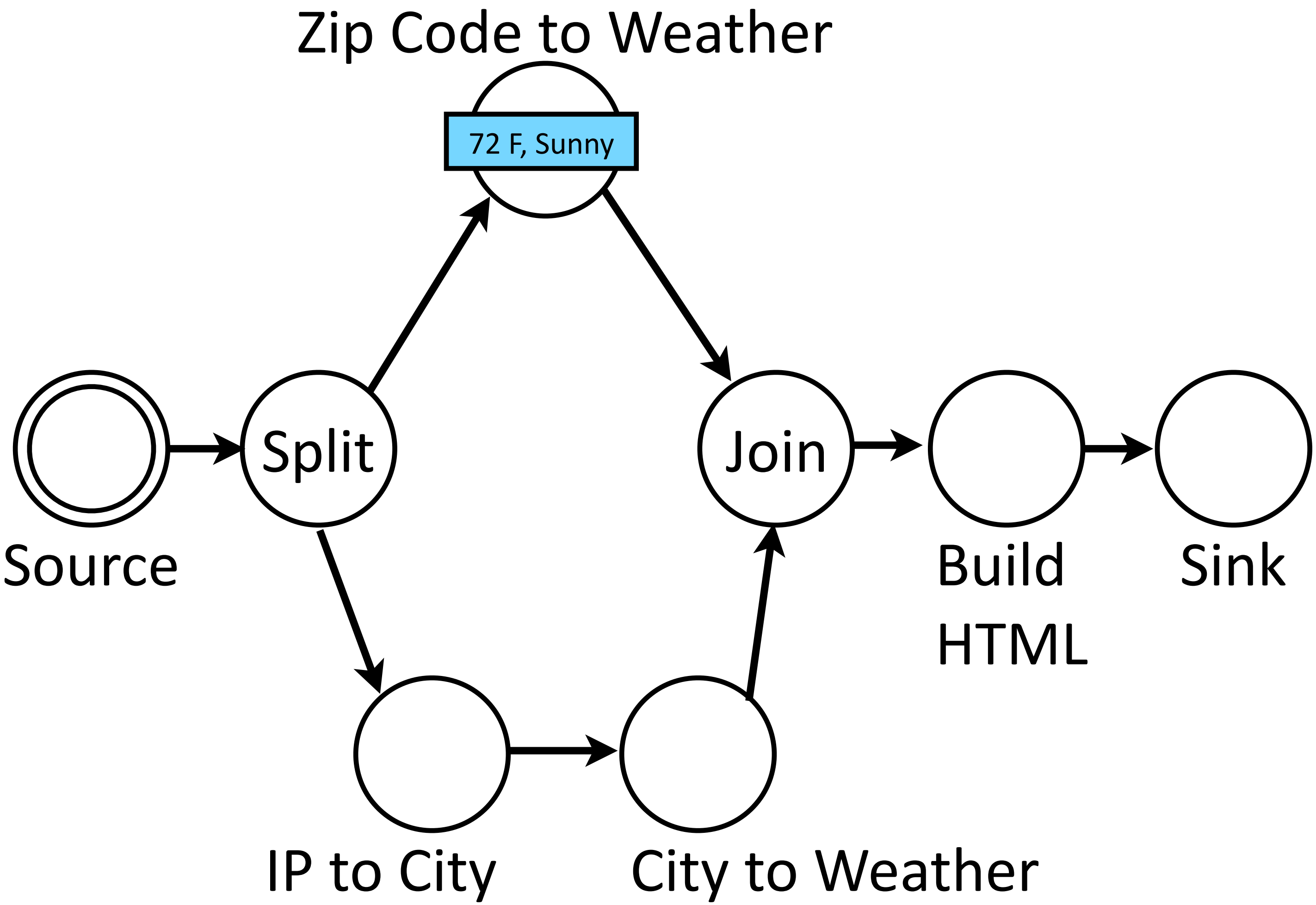
7

Weather Service - Sample Input

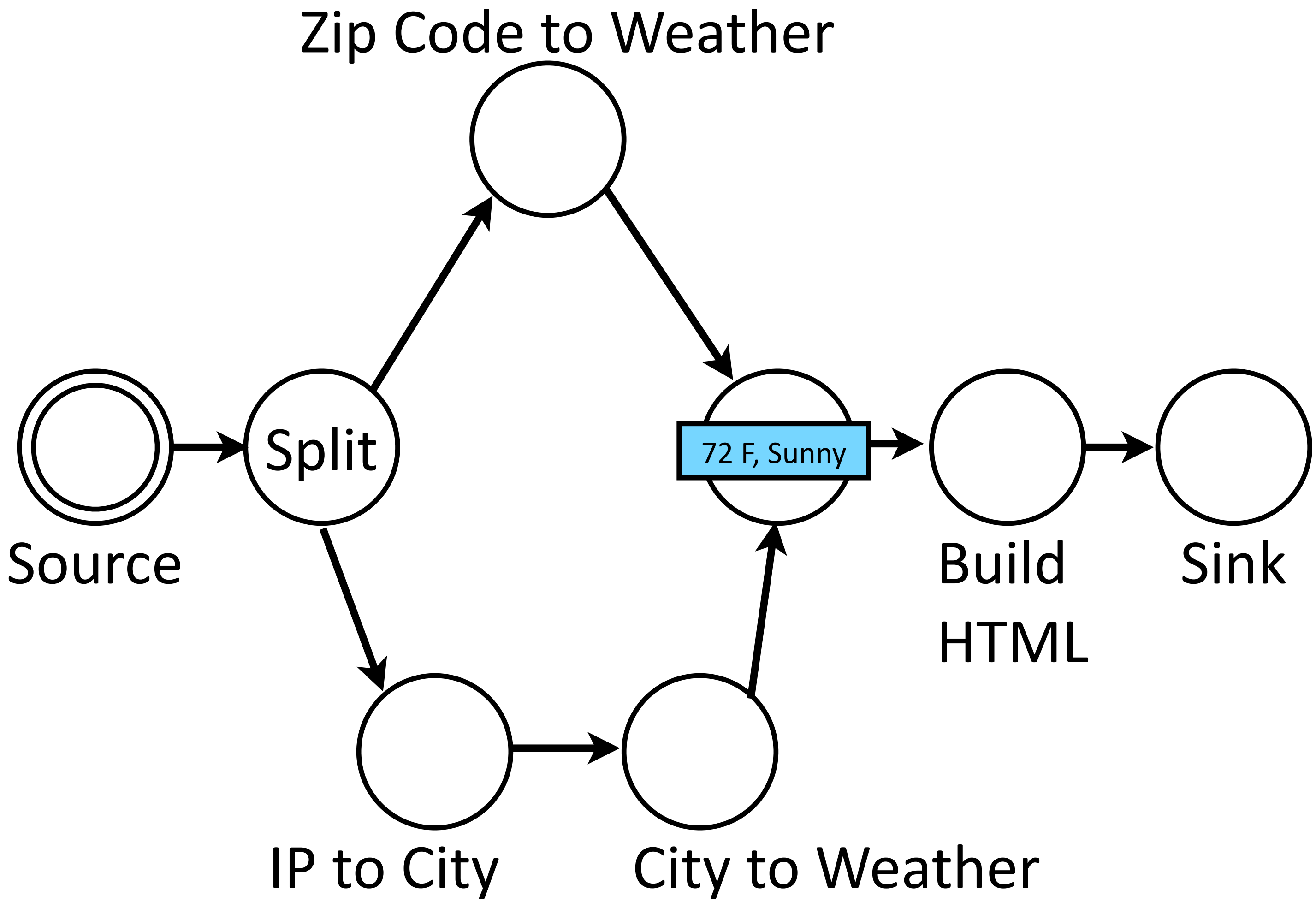


7

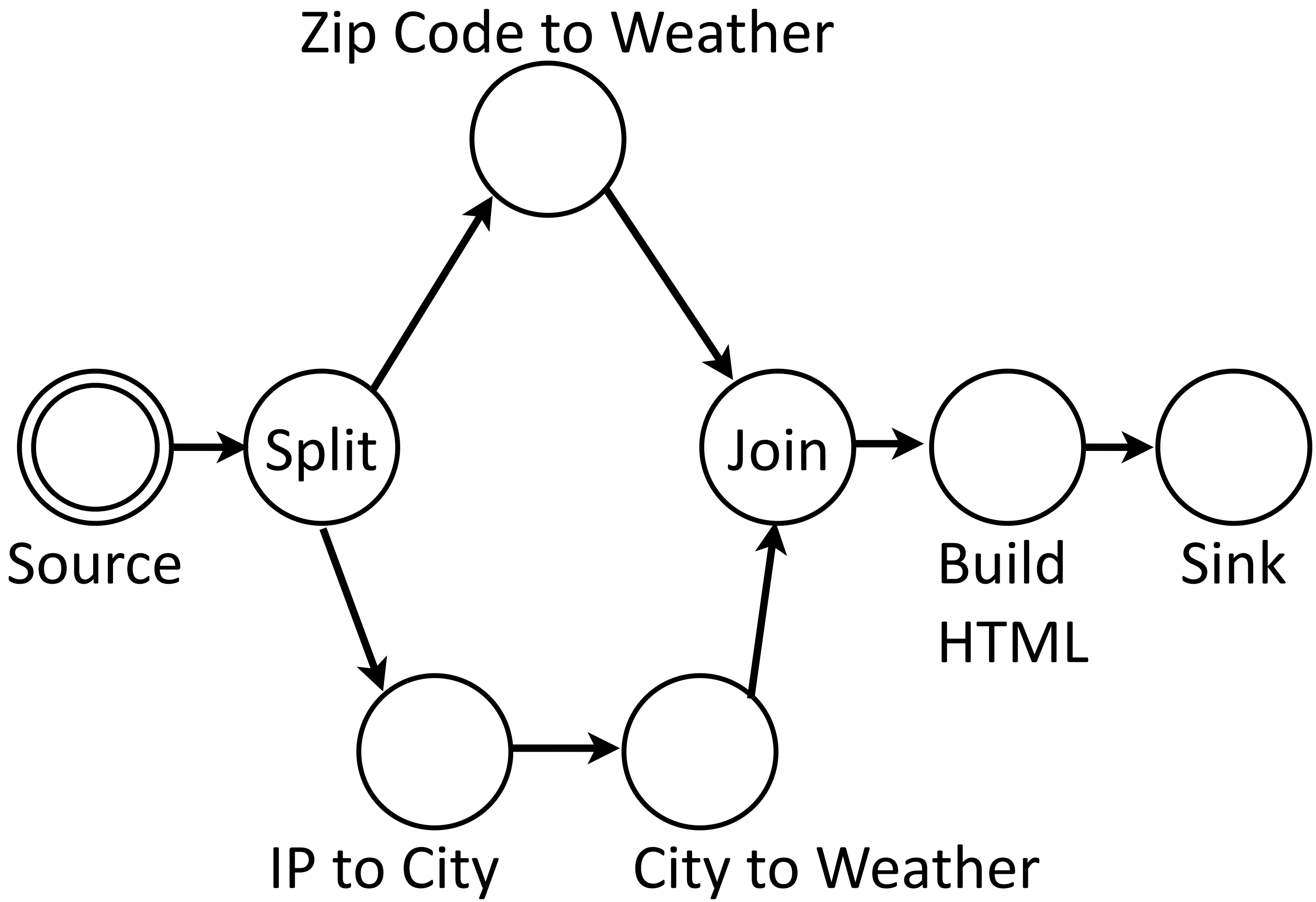
Weather Service - Sample Input



Weather Service - Sample Input

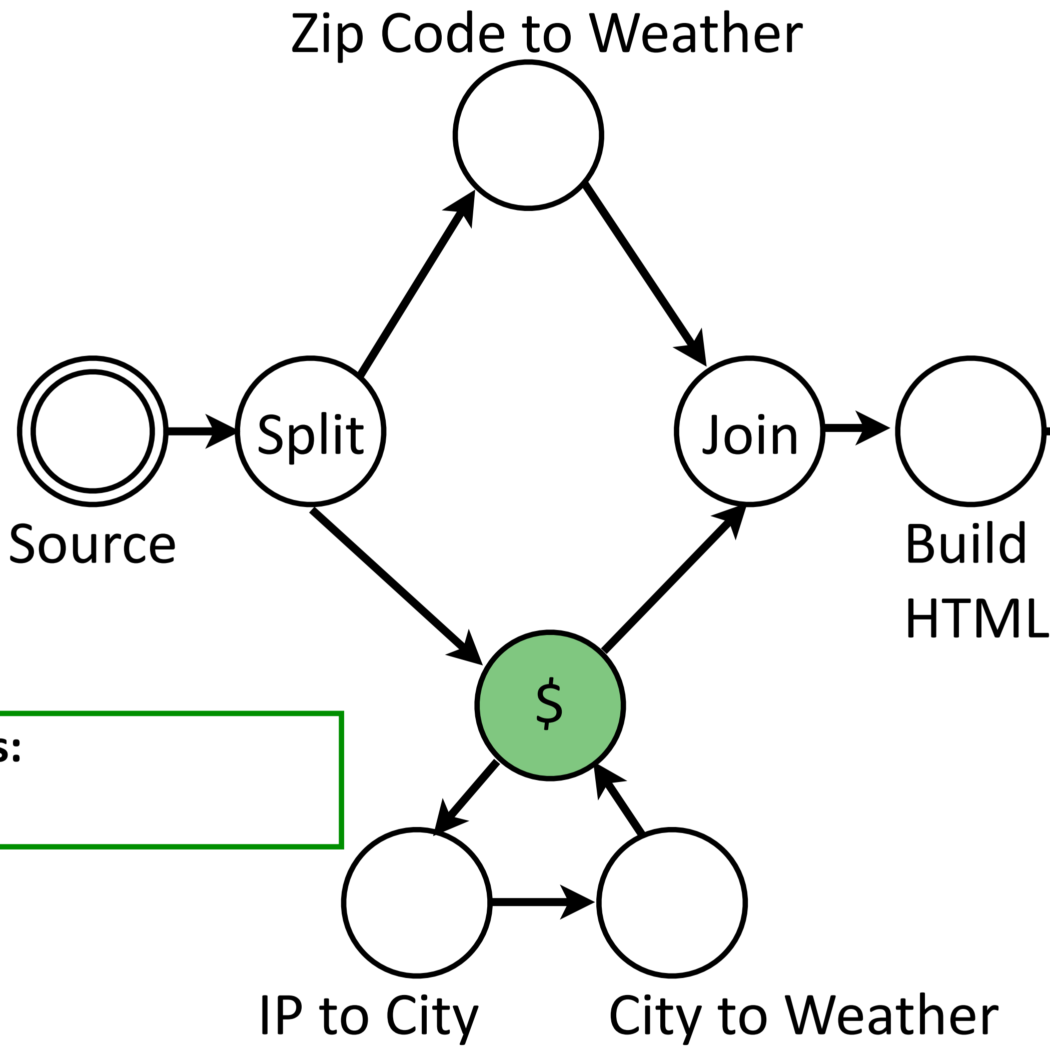


Weather Service - Sample Input



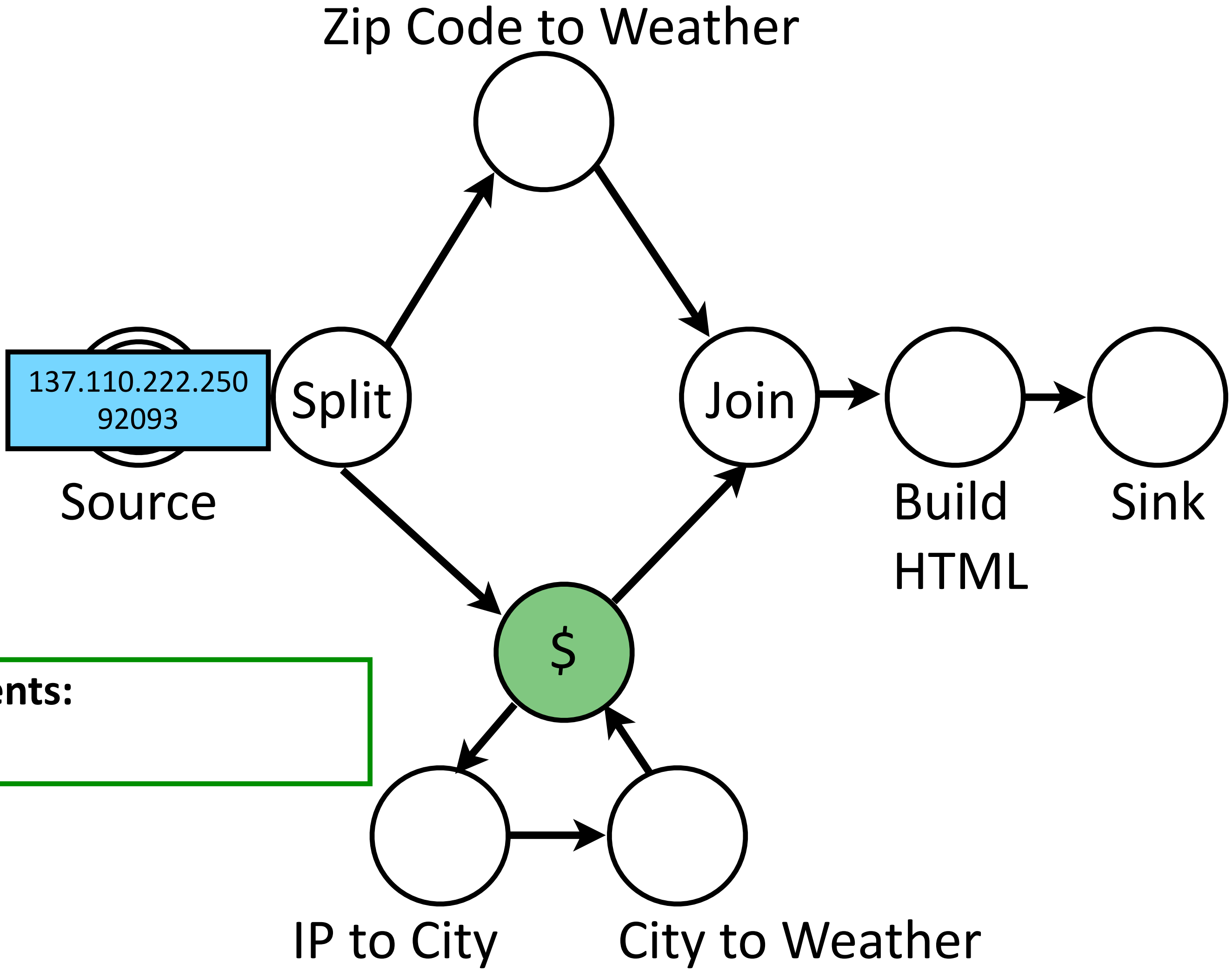
7

Caching {IP to City, City to Weather}



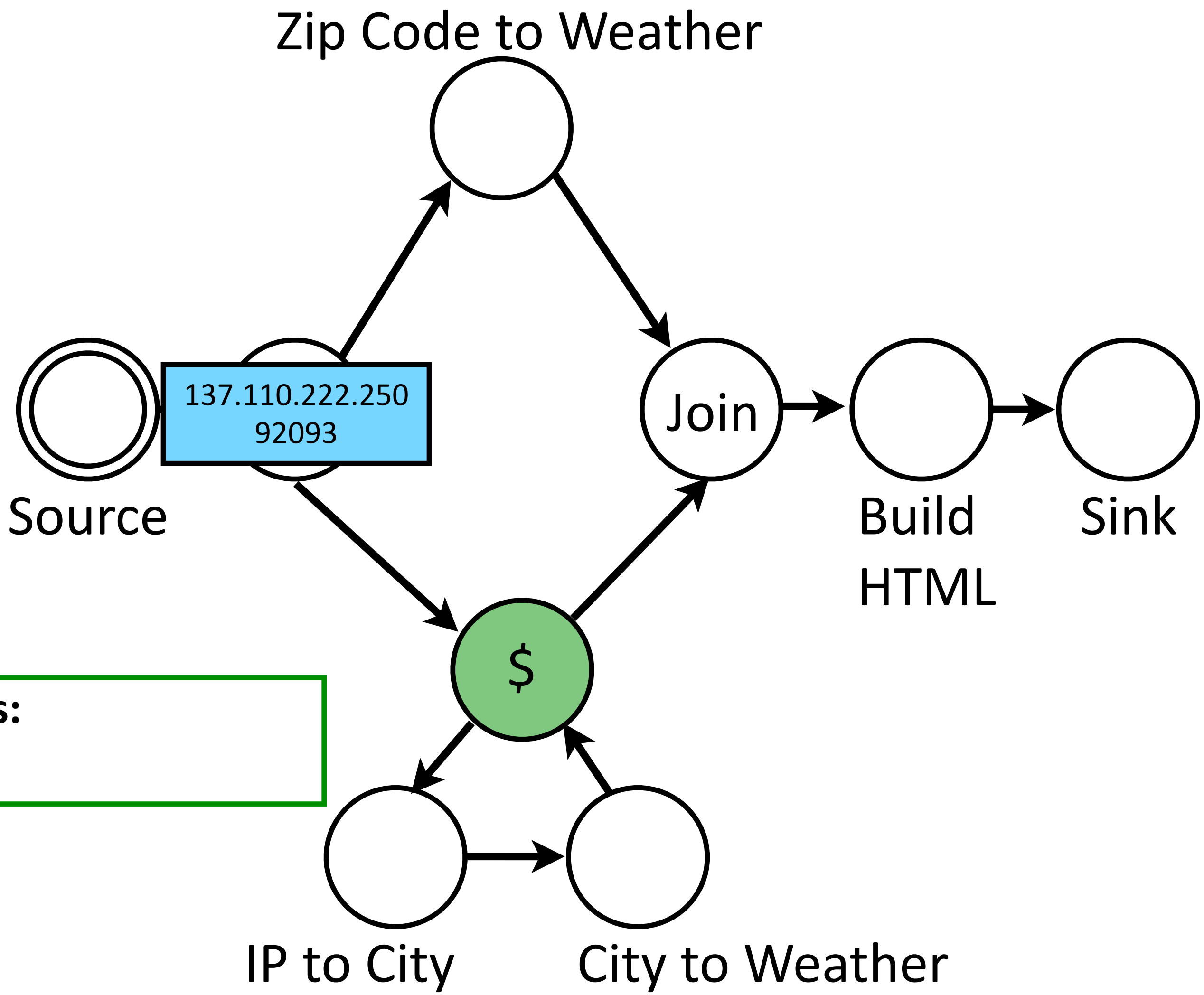
Cache Contents:

Caching {IP to City, City to Weather}



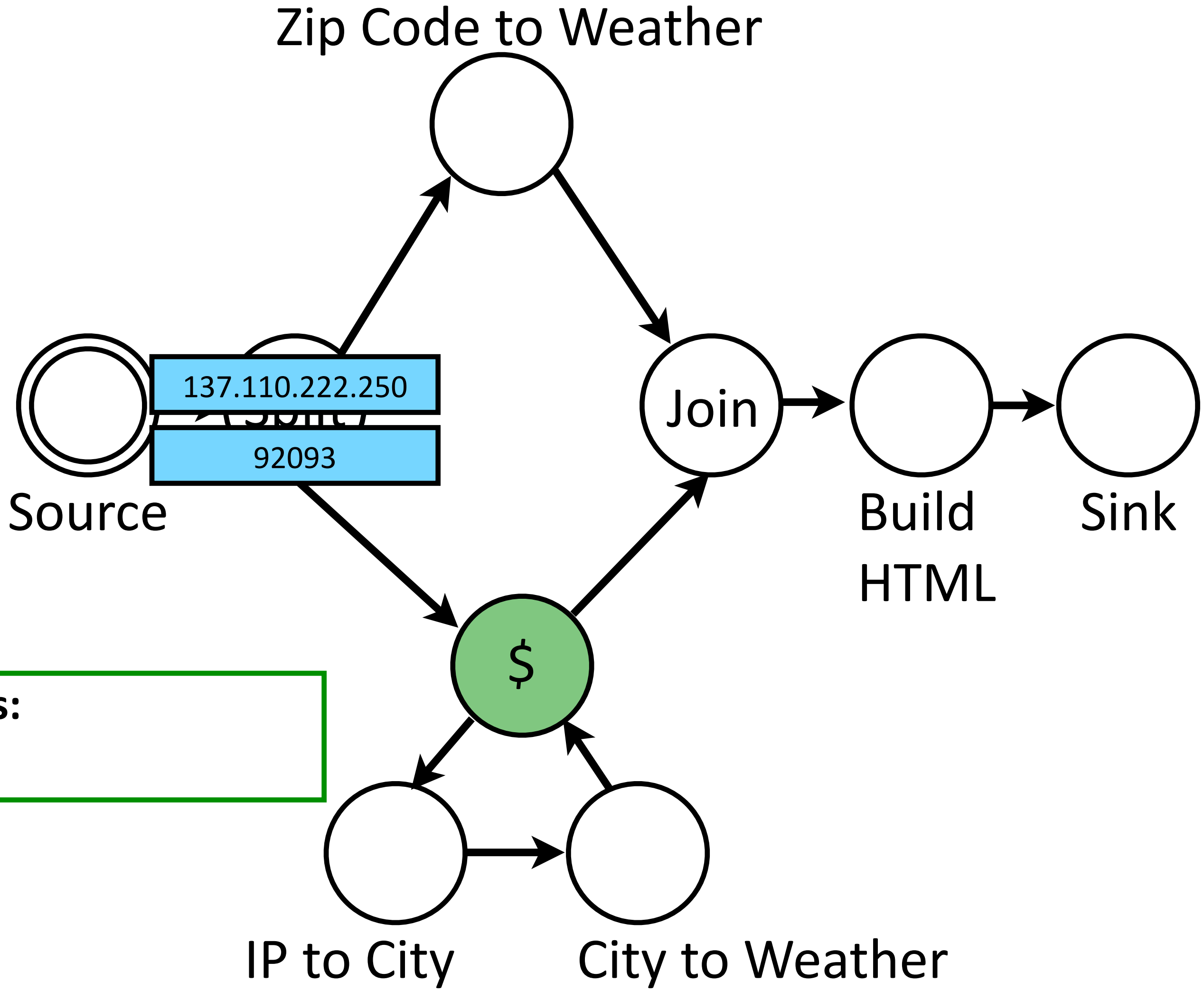
Cache Contents:

Caching {IP to City, City to Weather}



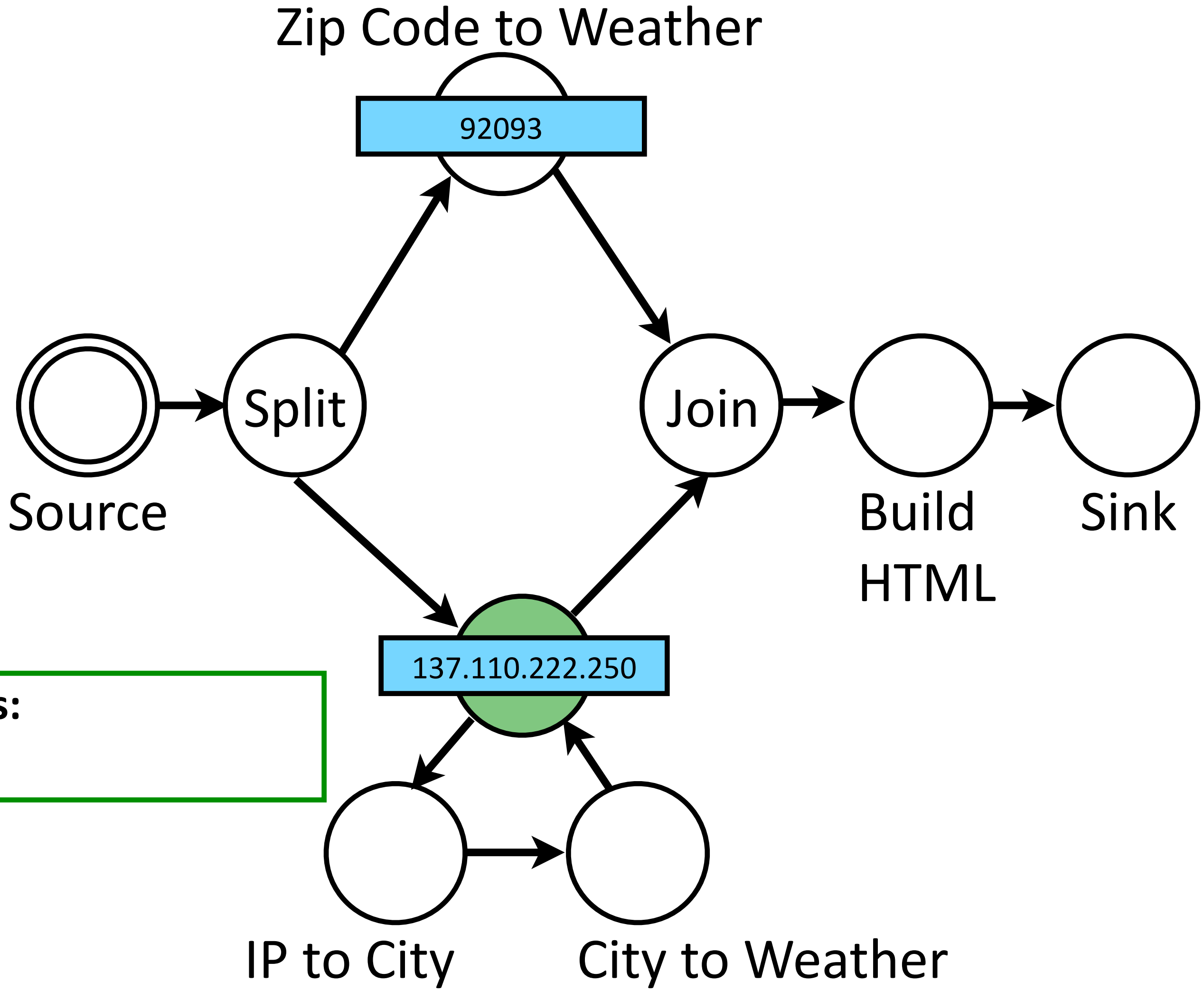
Cache Contents:

Caching {IP to City, City to Weather}



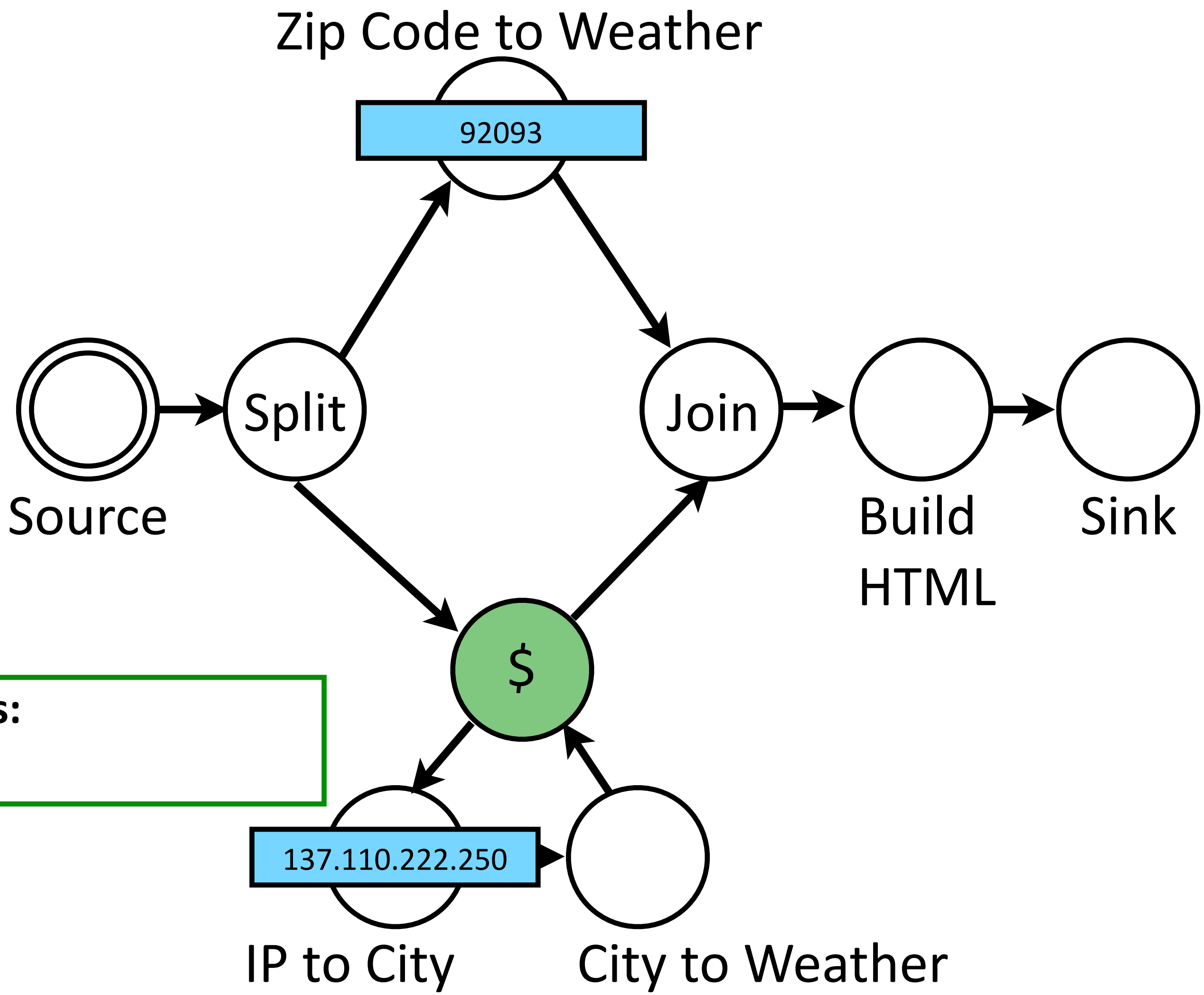
Cache Contents:

Caching {IP to City, City to Weather}



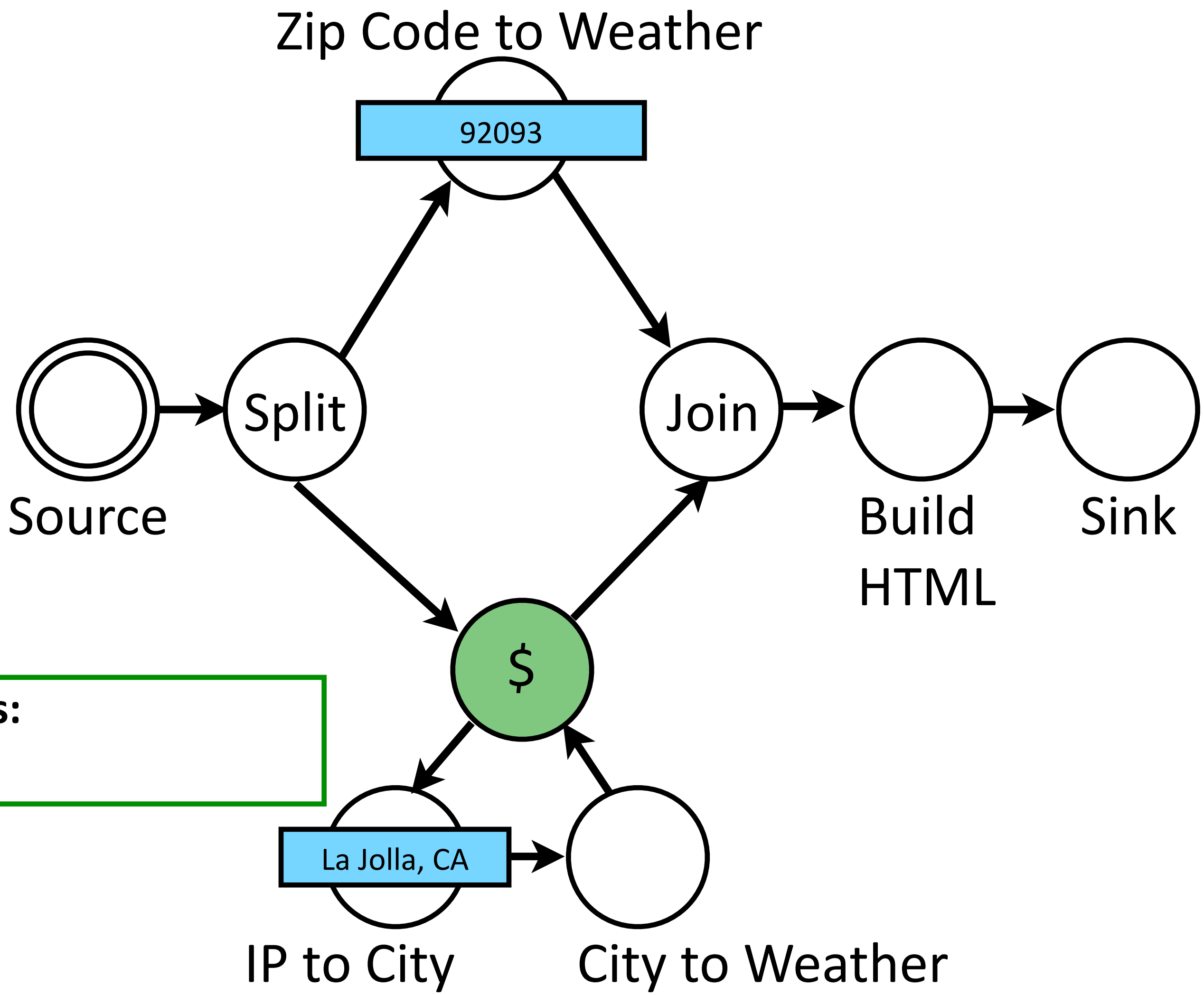
Cache Contents:

Caching {IP to City, City to Weather}



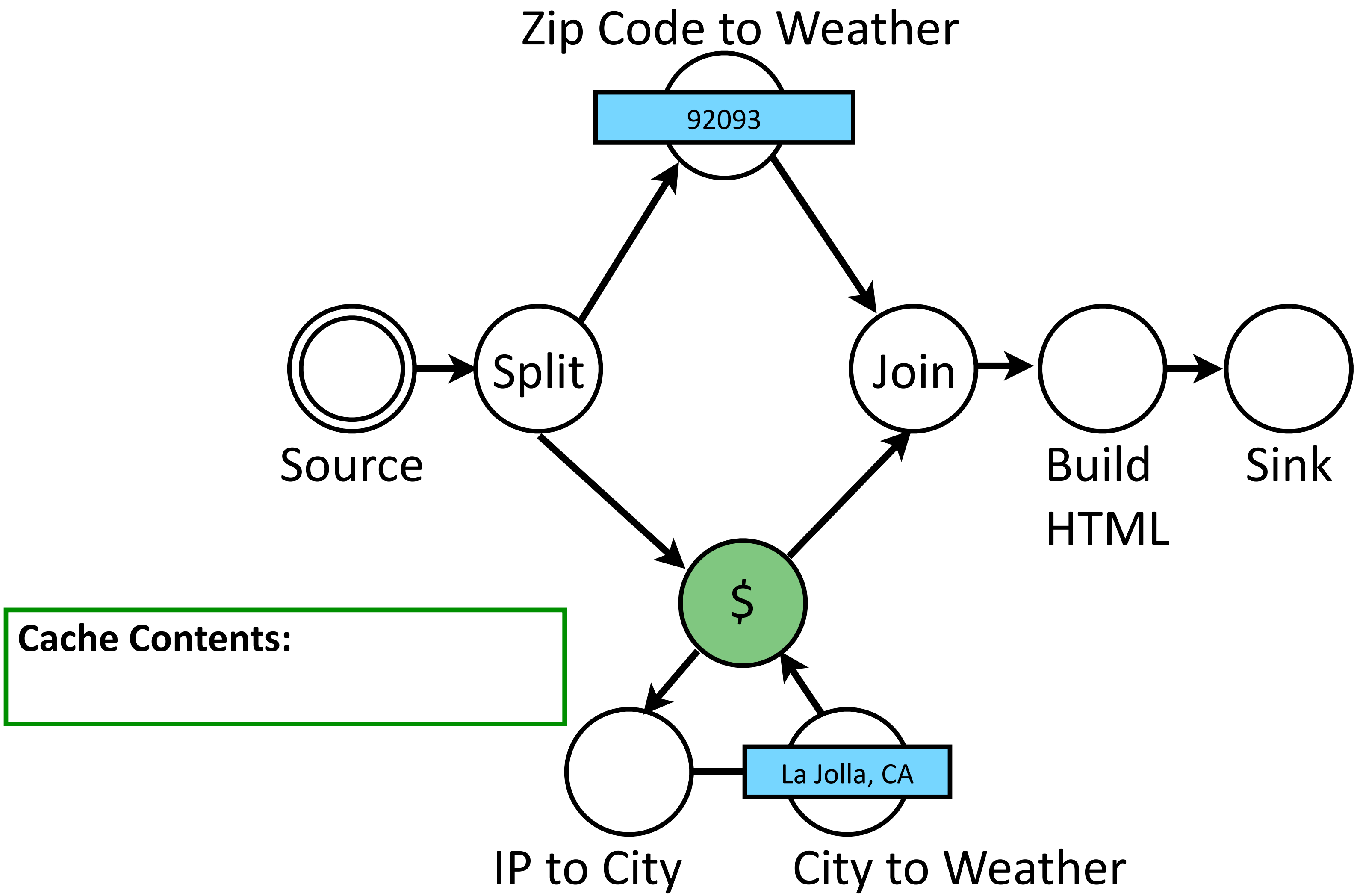
Cache Contents:

Caching {IP to City, City to Weather}

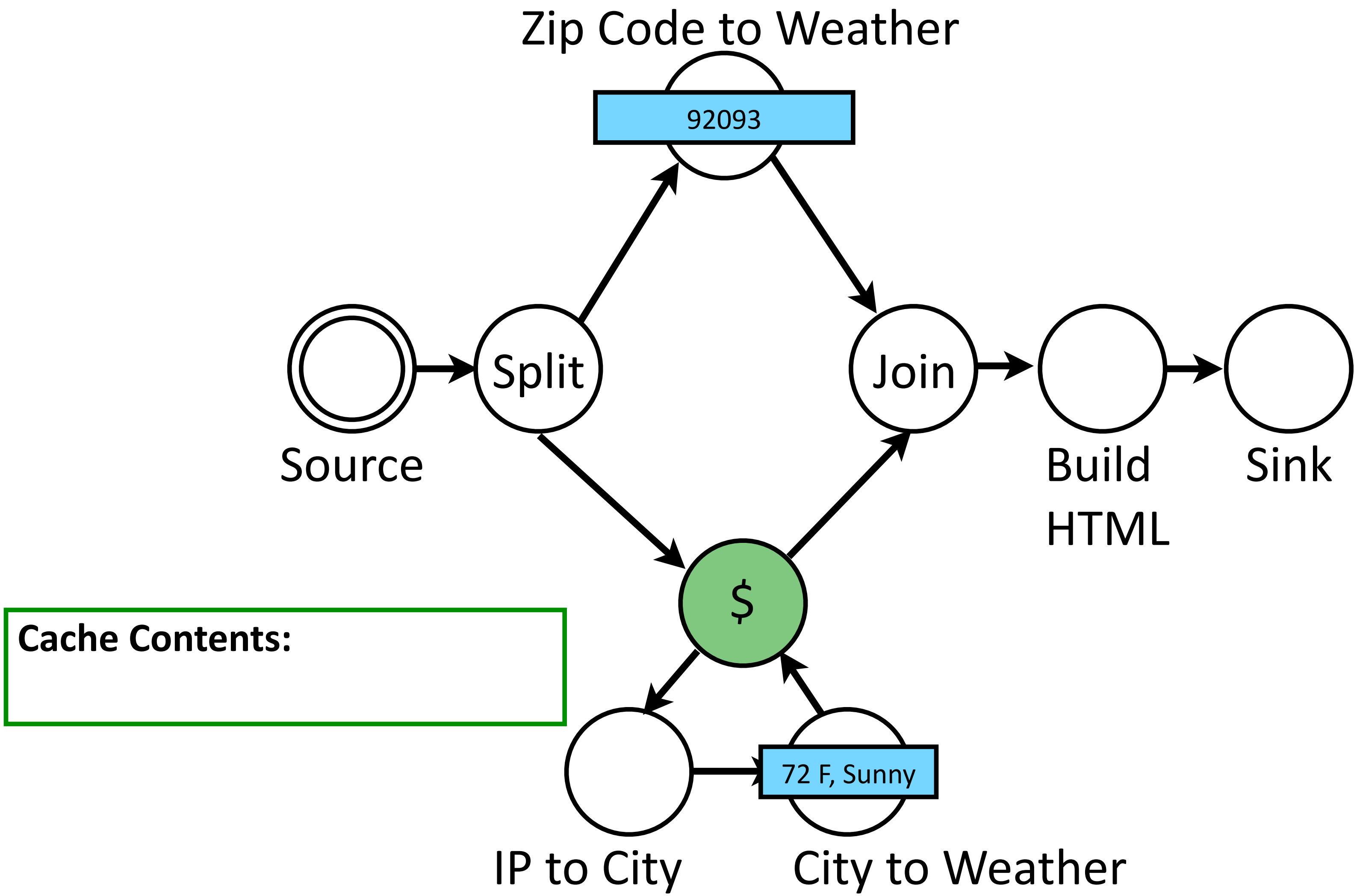


Cache Contents:

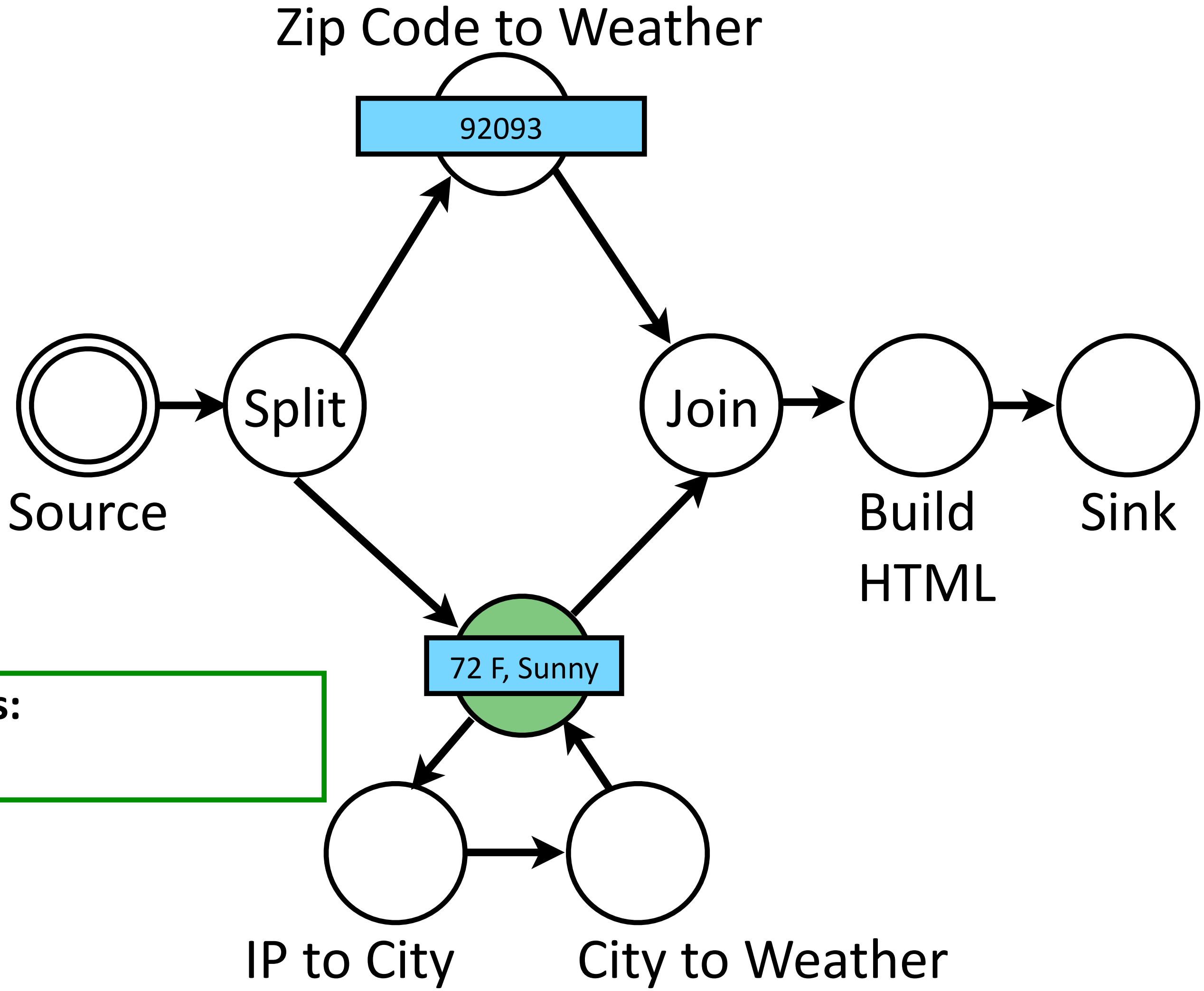
Caching {IP to City, City to Weather}



Caching {IP to City, City to Weather}

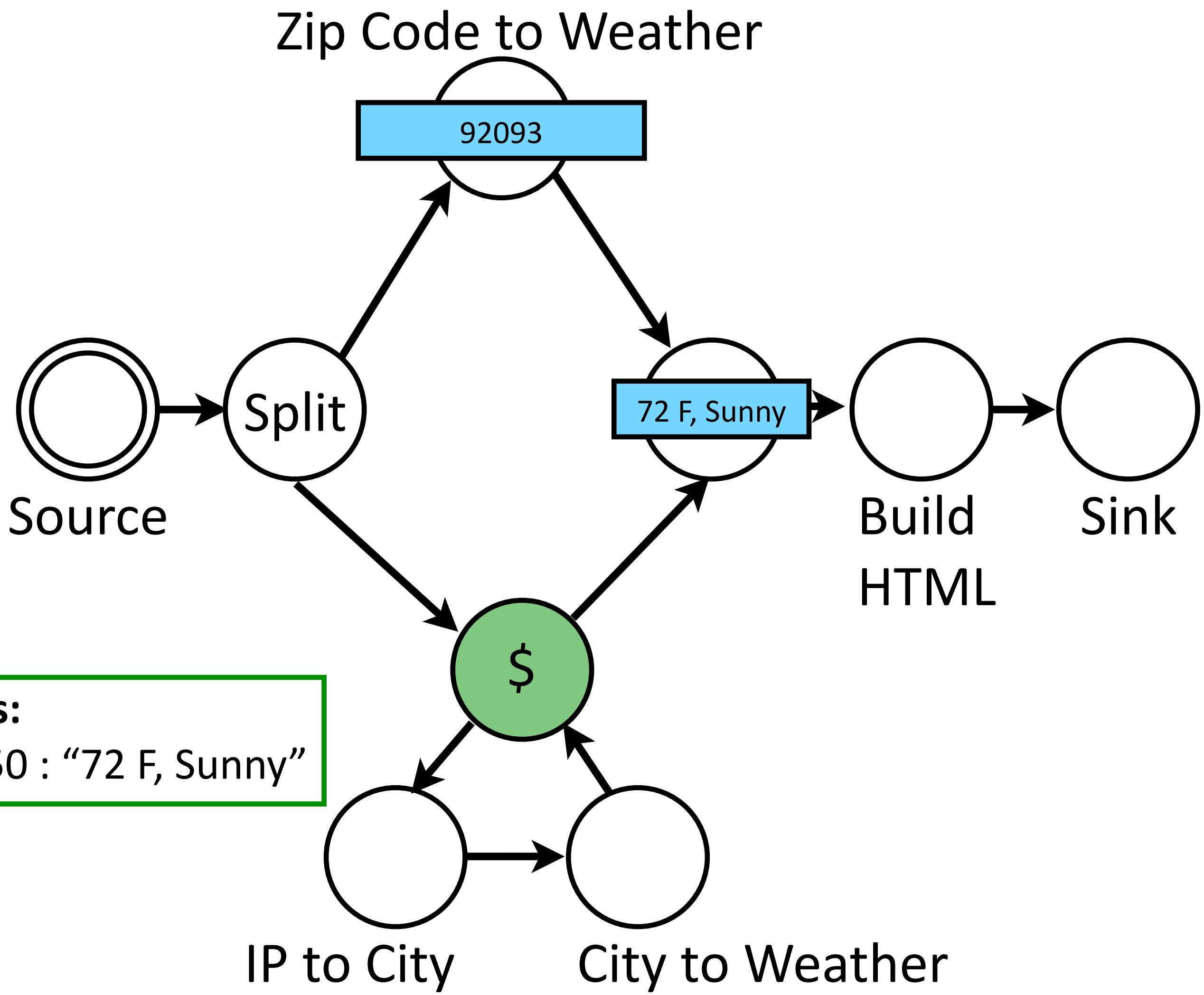


Caching {IP to City, City to Weather}



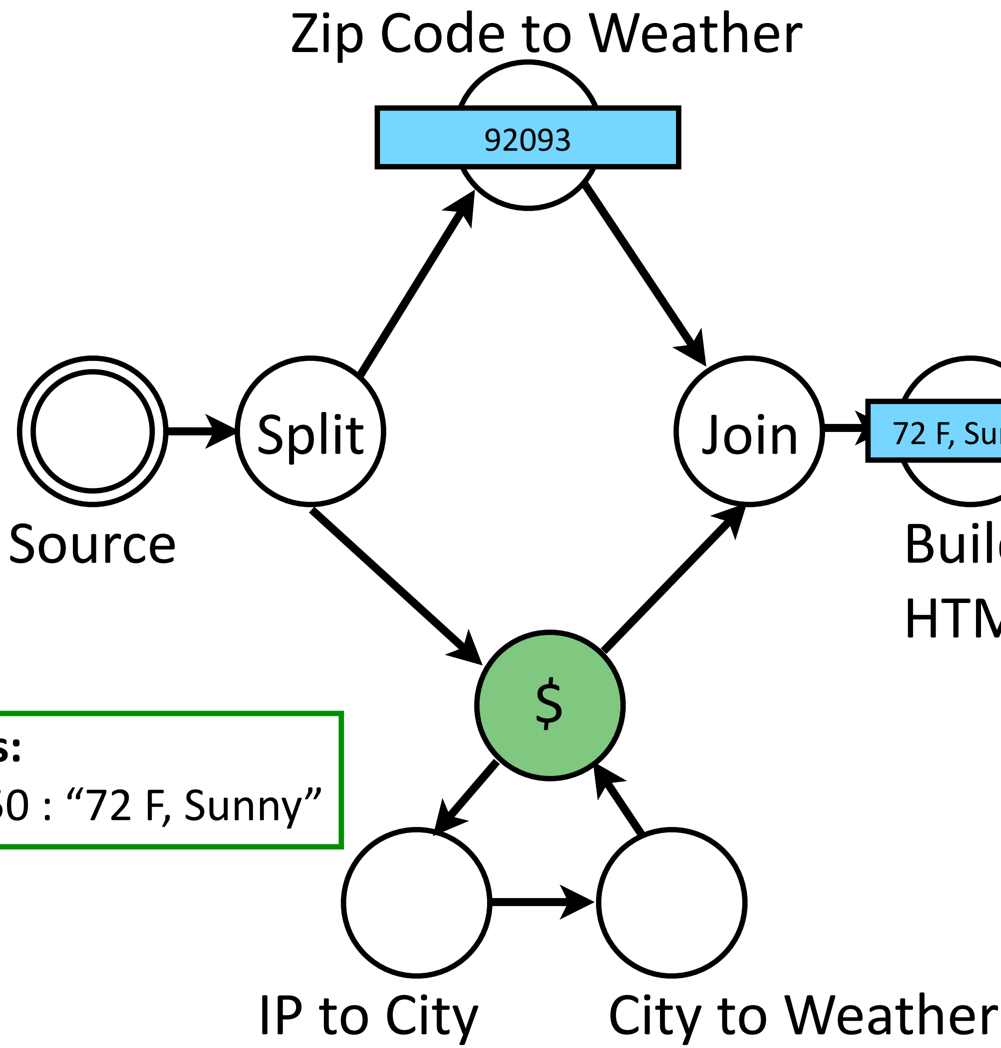
Cache Contents:

Caching {IP to City, City to Weather}



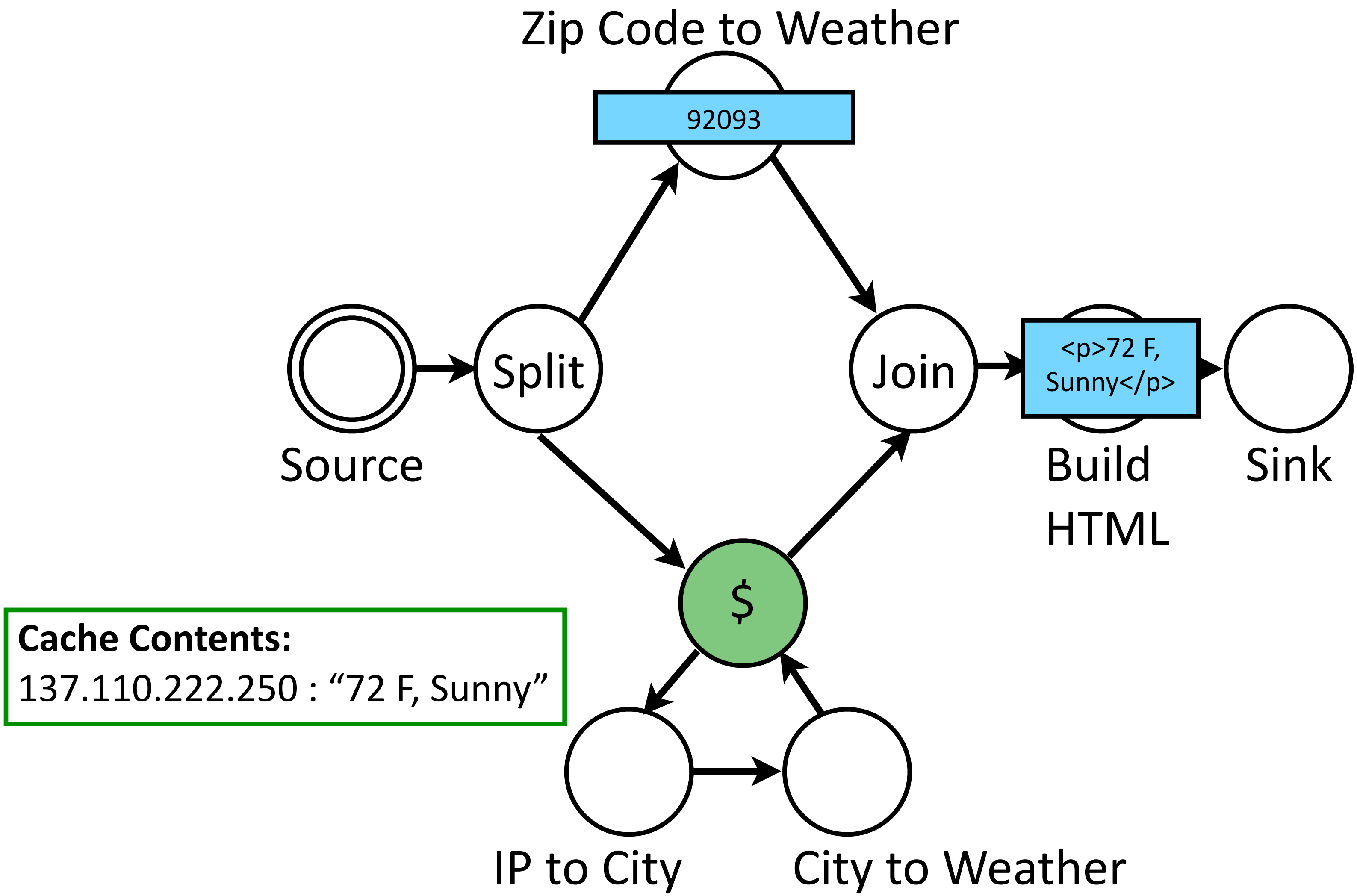
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



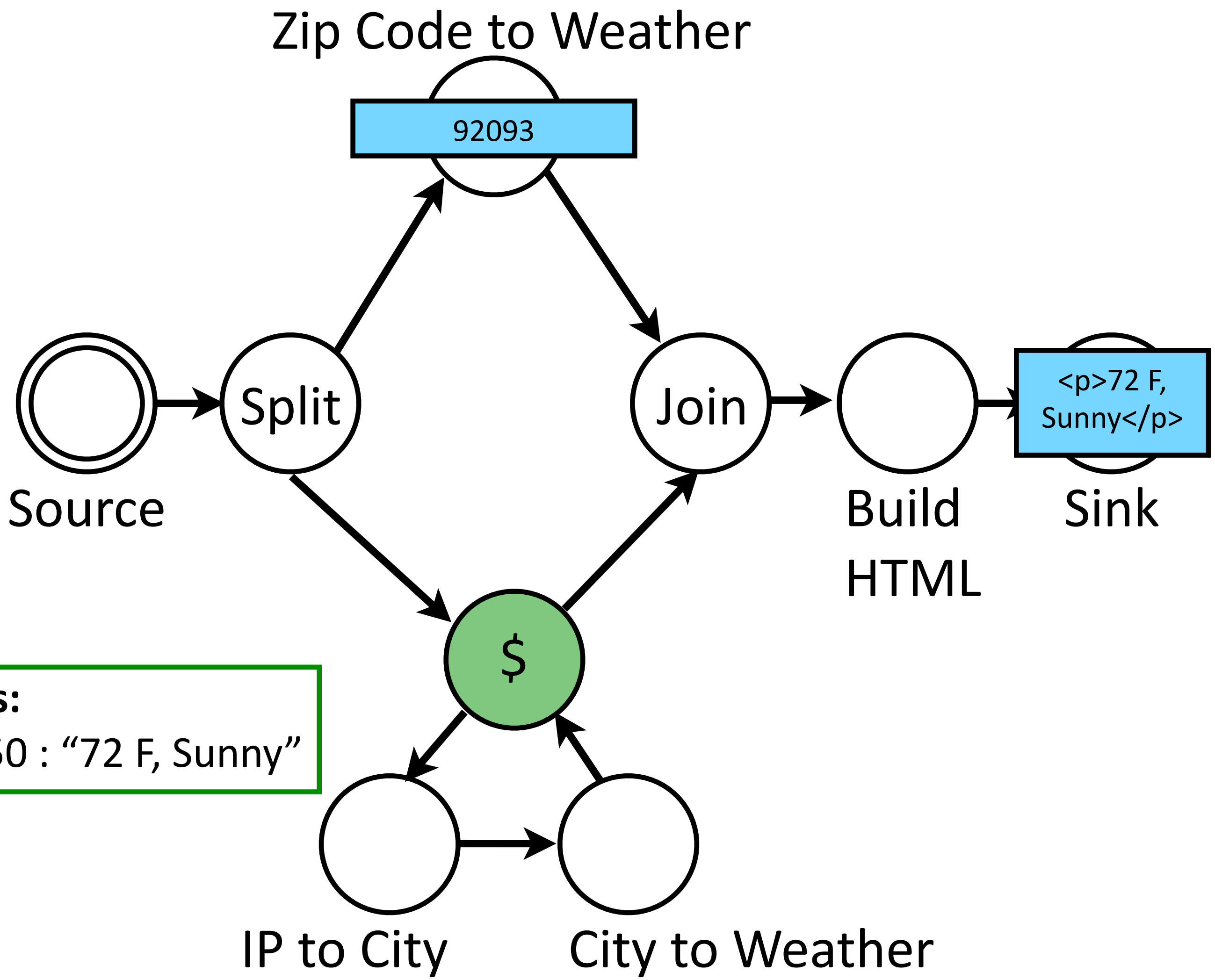
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



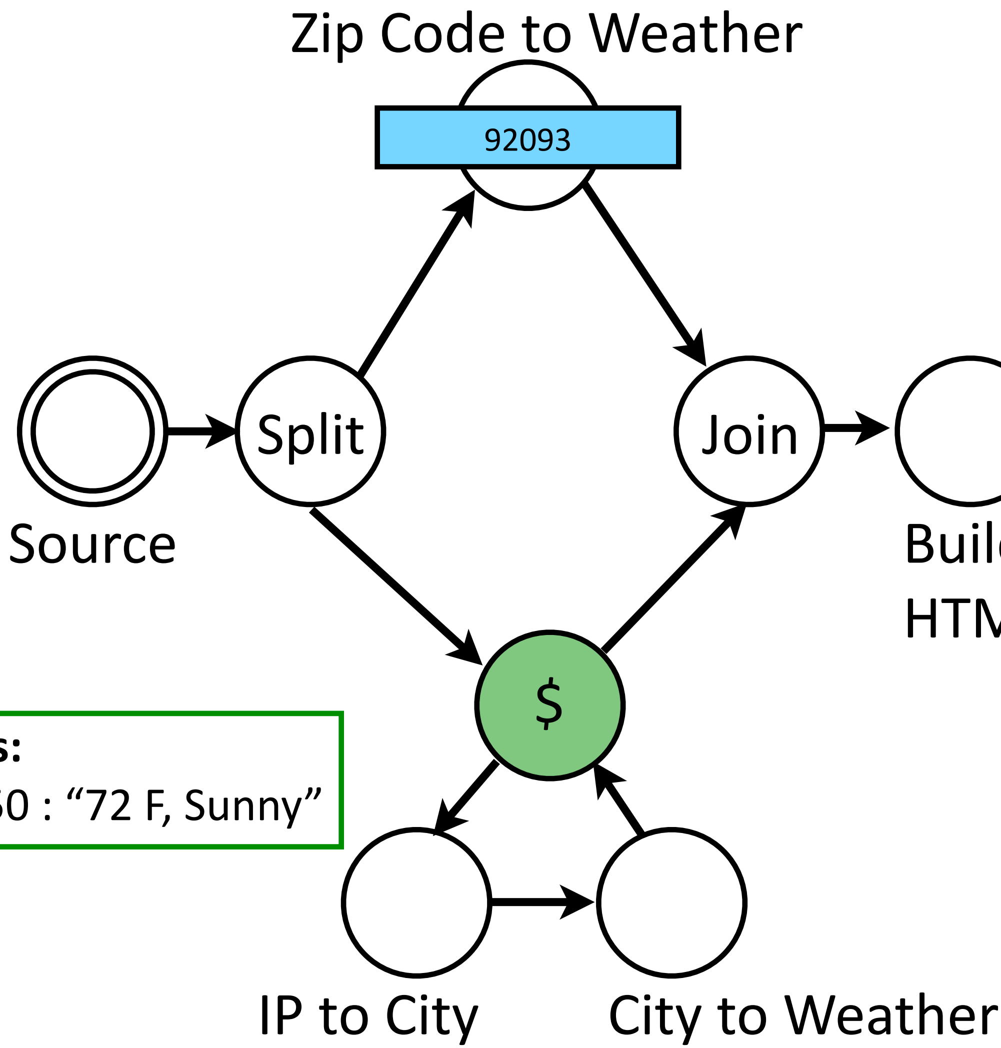
8

Caching {IP to City, City to Weather}



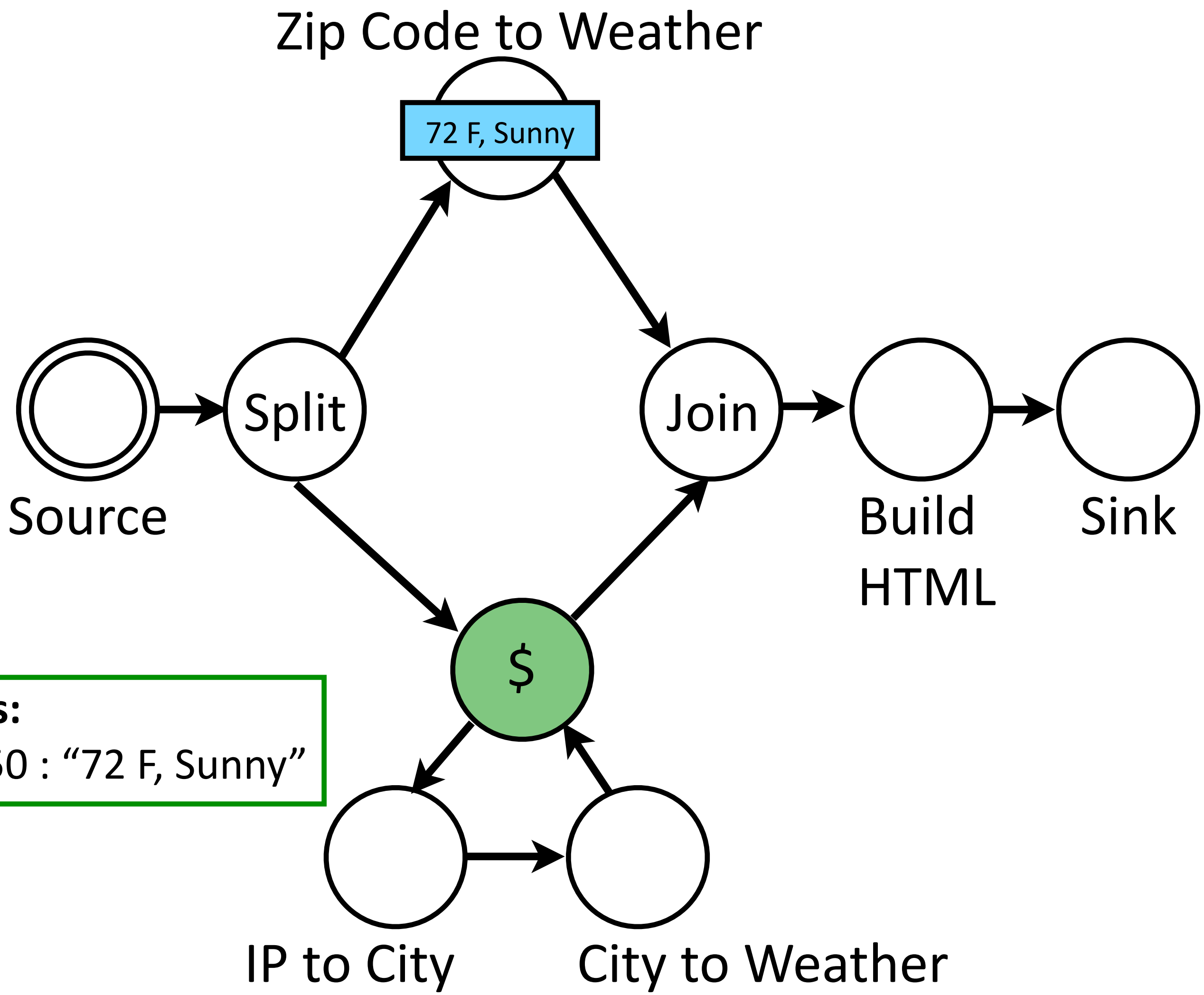
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



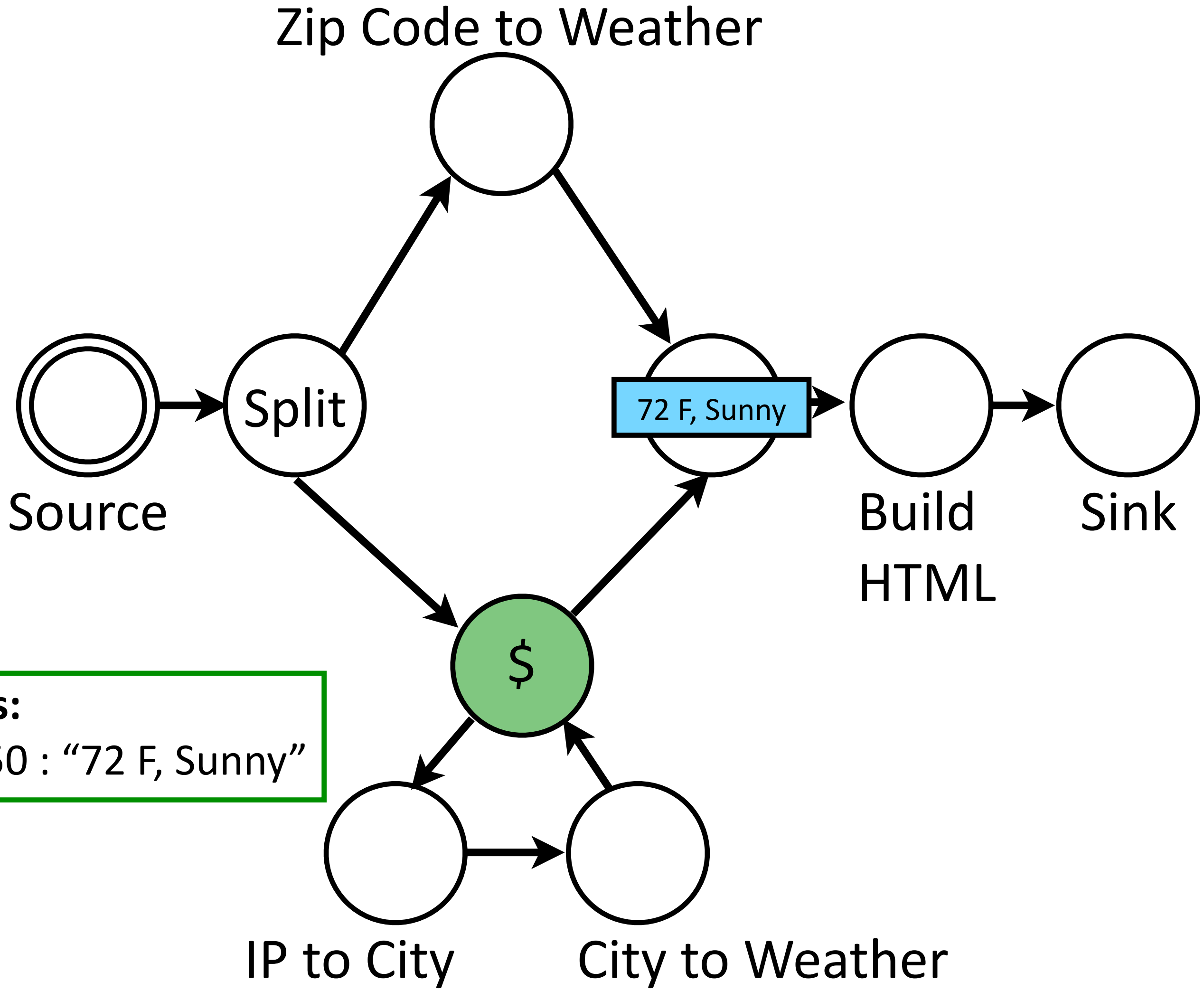
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



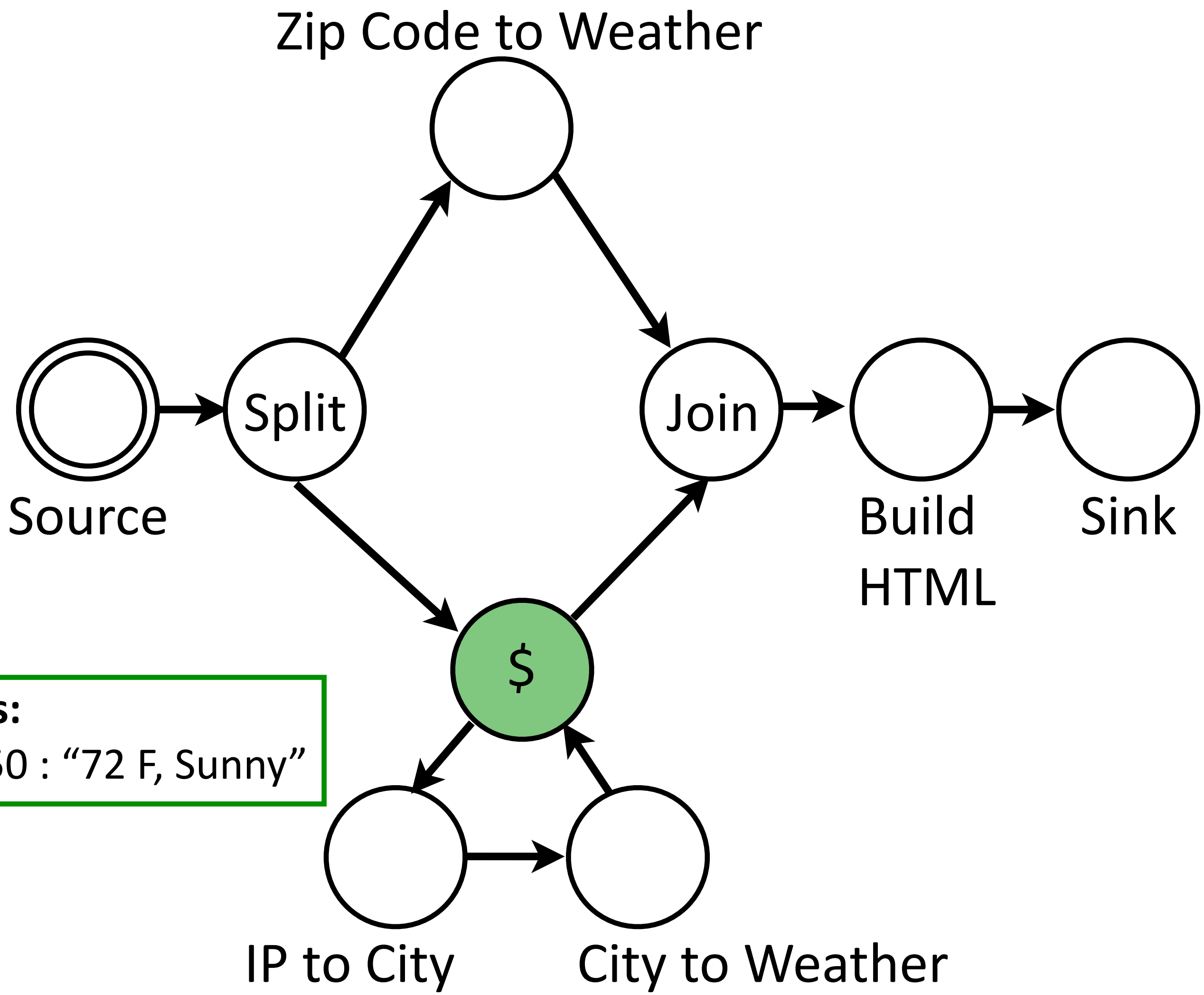
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



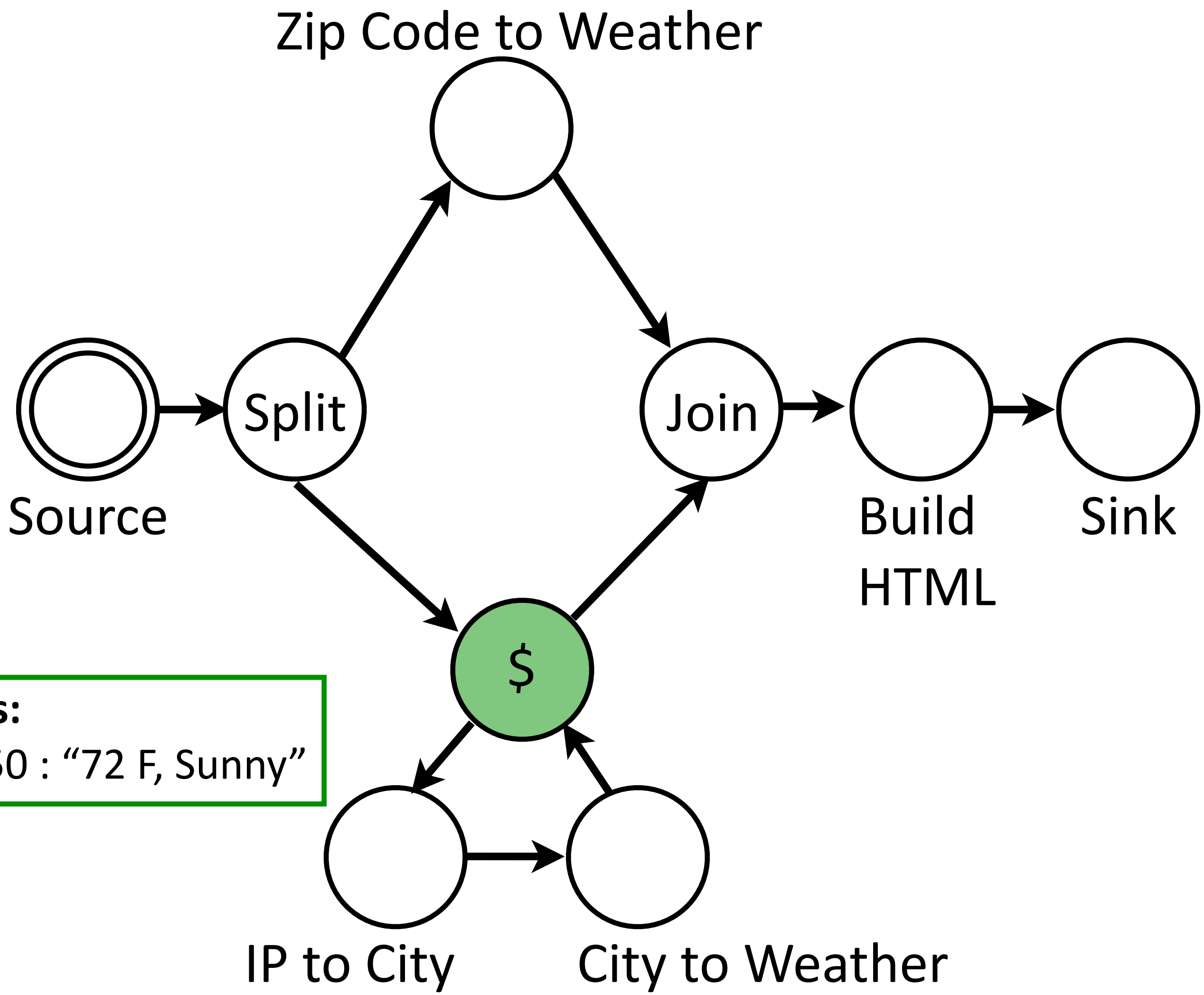
Cache Contents:
137.110.222.250 : "72 F, Sunny"

Caching {IP to City, City to Weather}



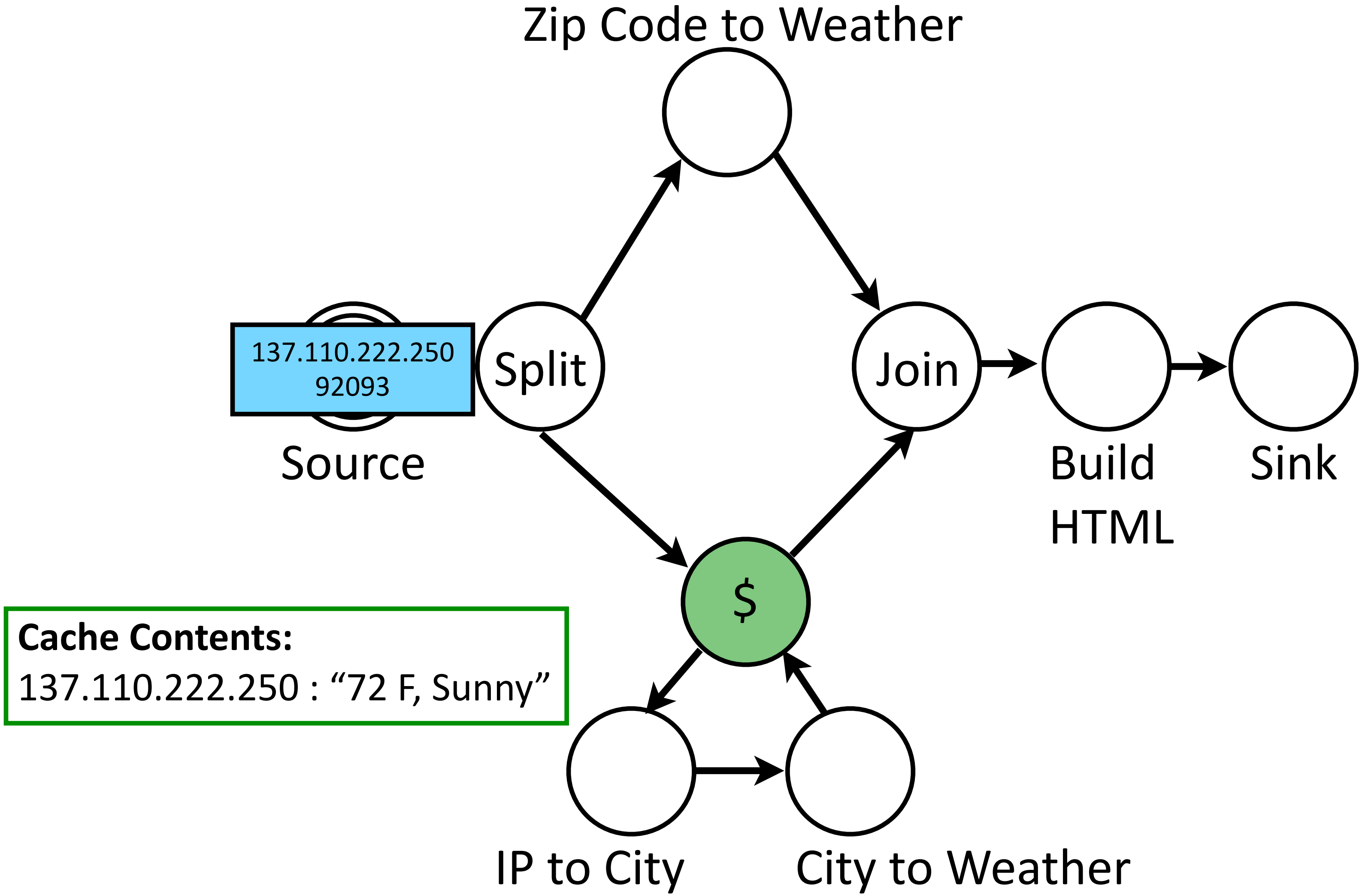
Cache Contents:
137.110.222.250 : "72 F, Sunny"

Caching {IP to City, City to Weather}

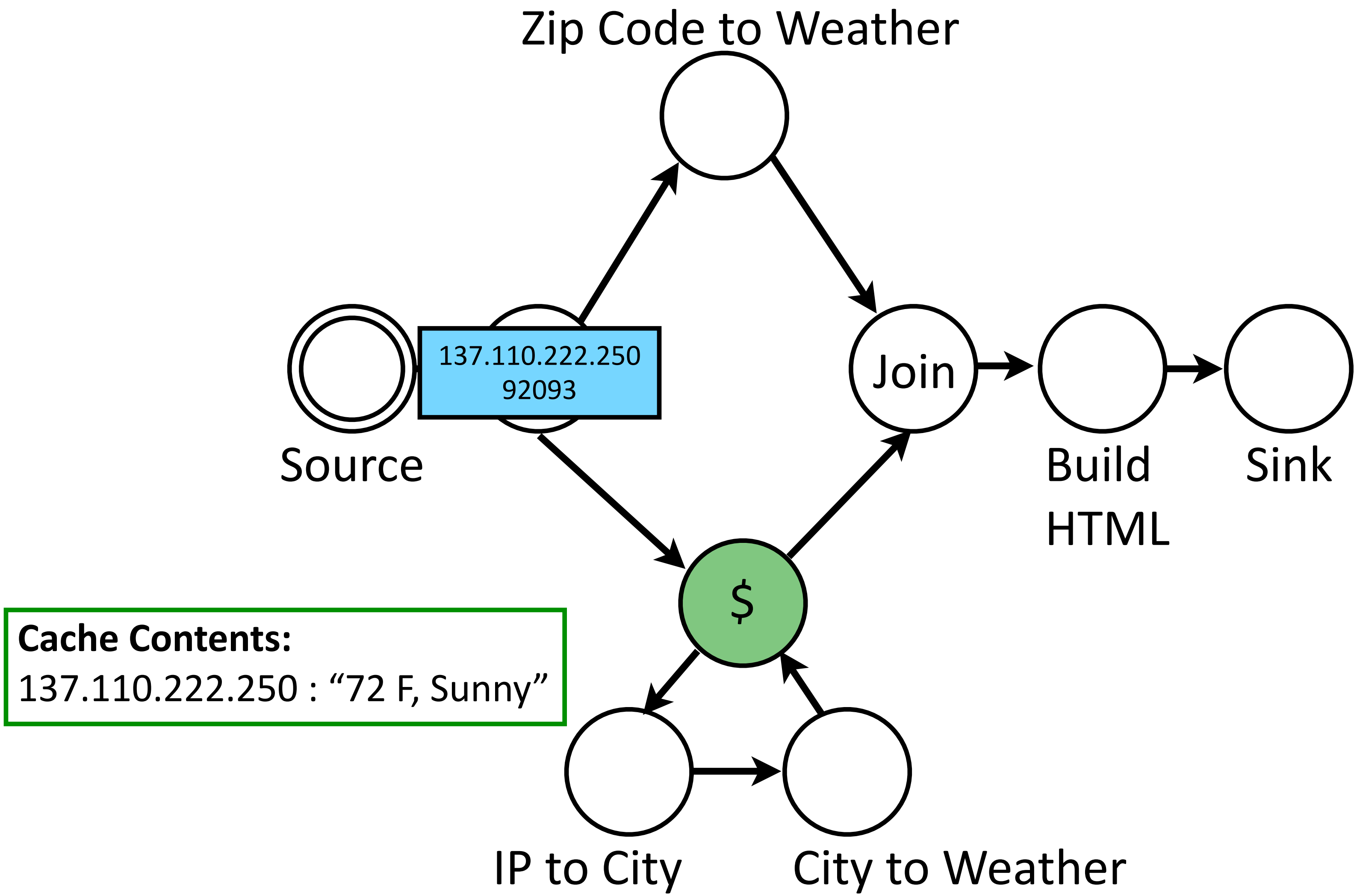


Cache Contents:
137.110.222.250 : “72 F, Sunny”

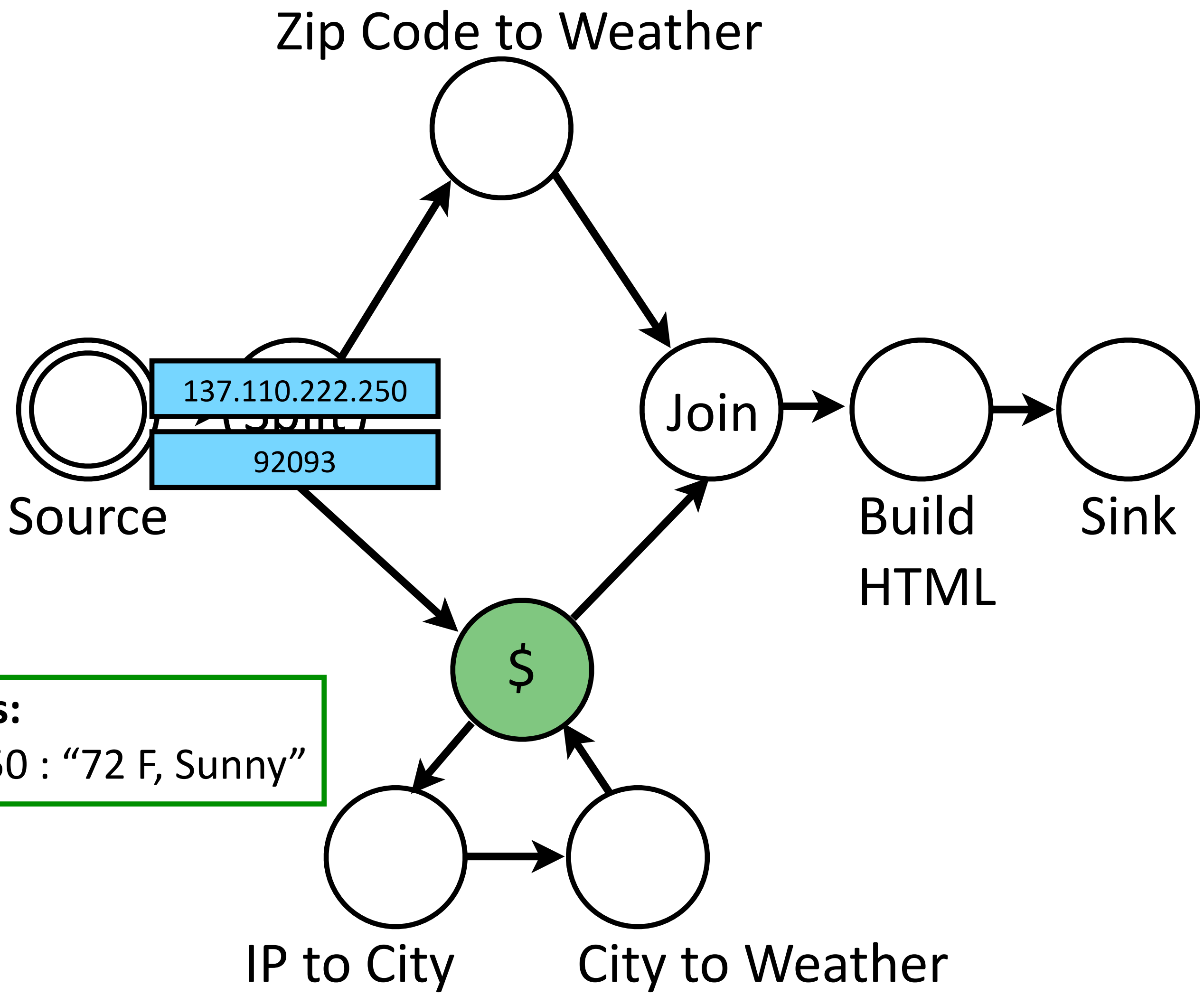
Caching {IP to City, City to Weather}



Caching {IP to City, City to Weather}

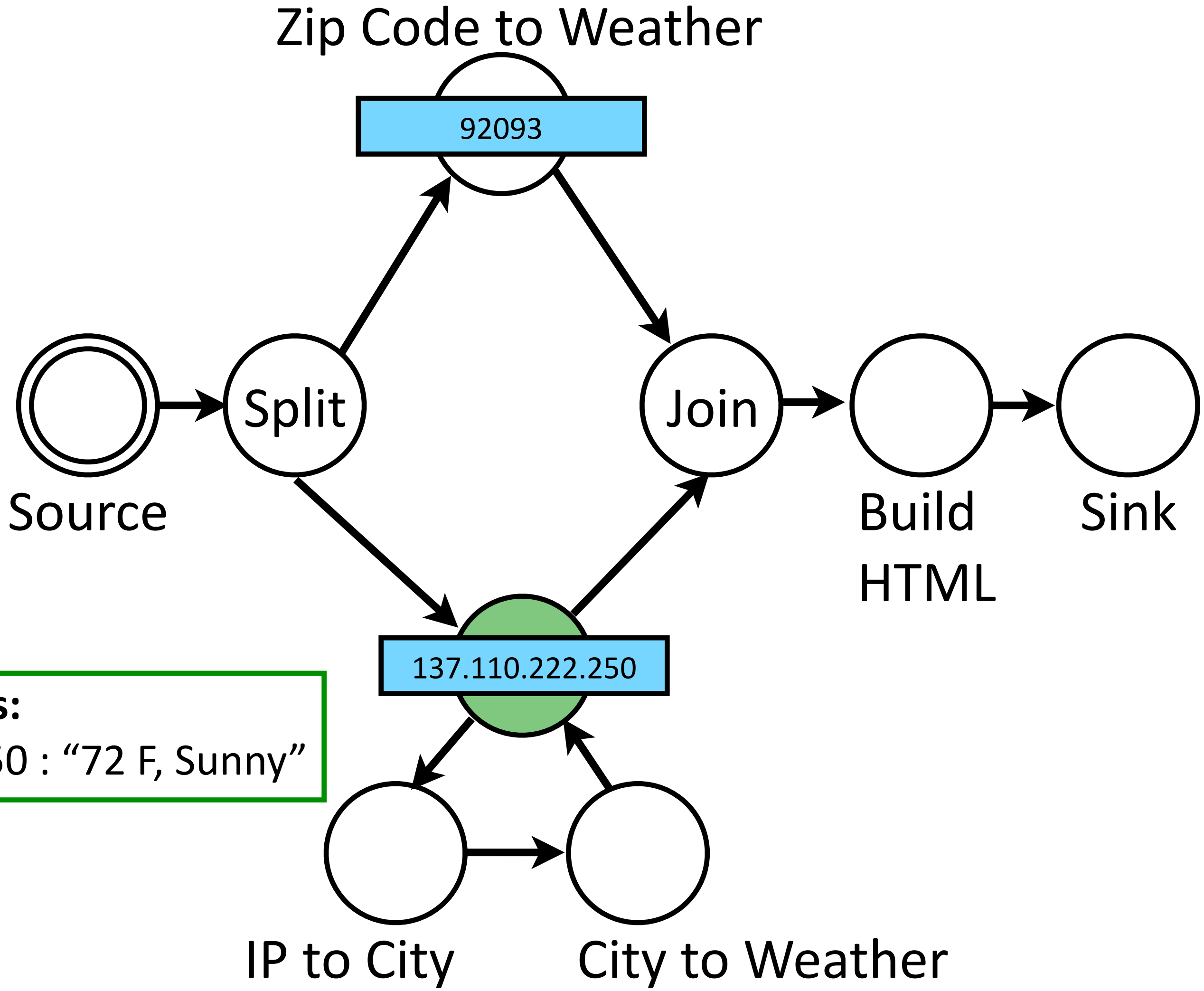


Caching {IP to City, City to Weather}



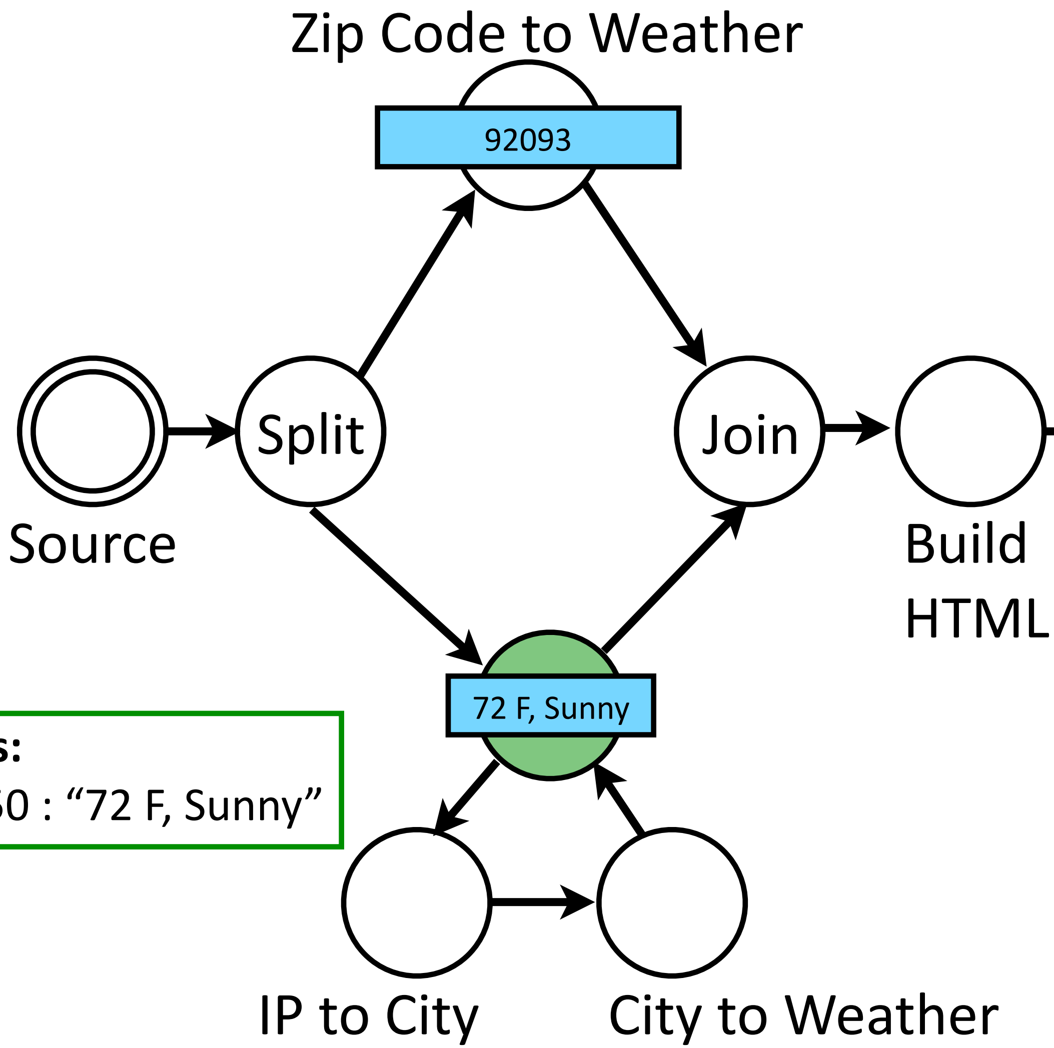
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



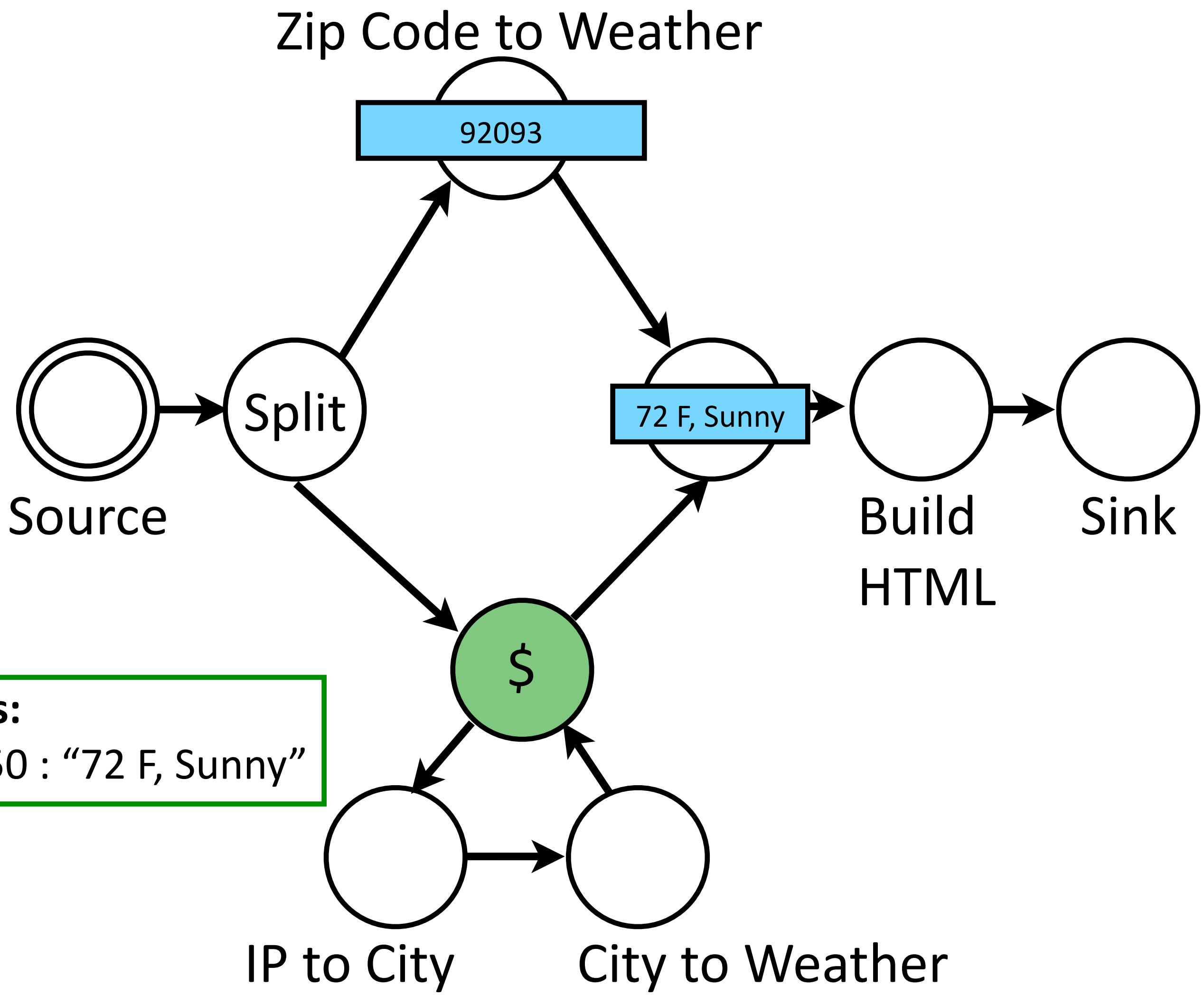
Cache Contents:
137.110.222.250 : "72 F, Sunny"

Caching {IP to City, City to Weather}



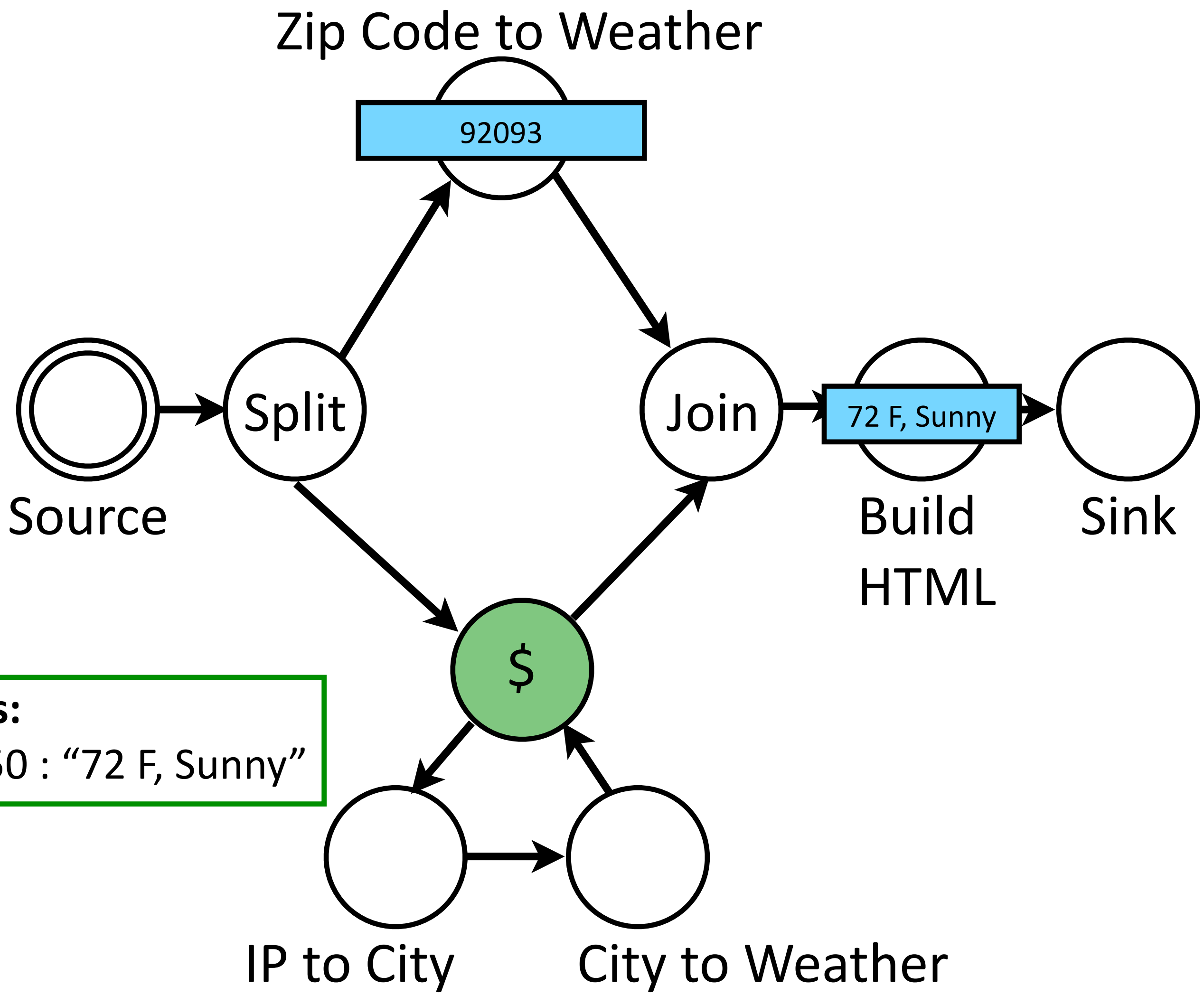
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



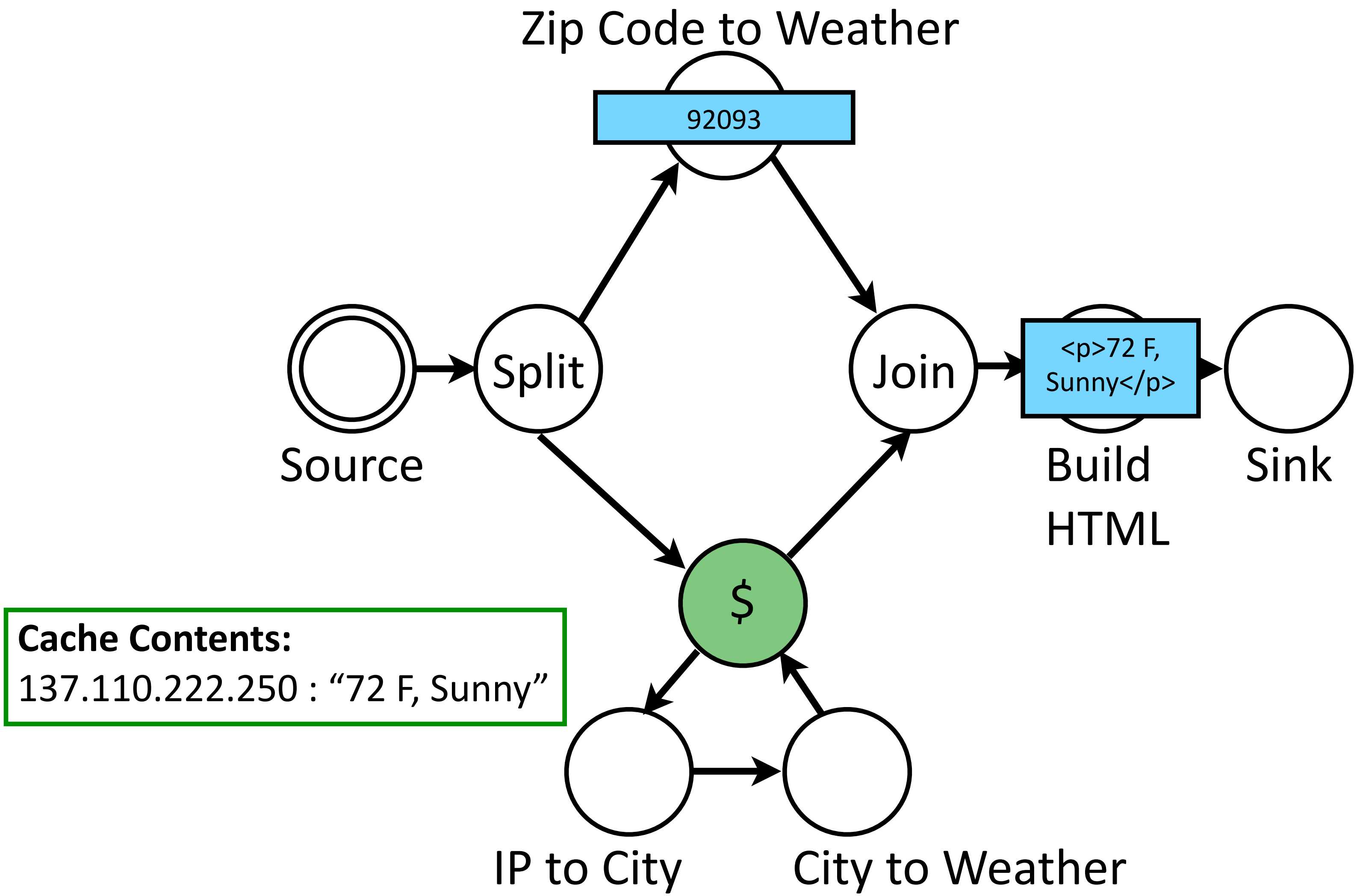
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}

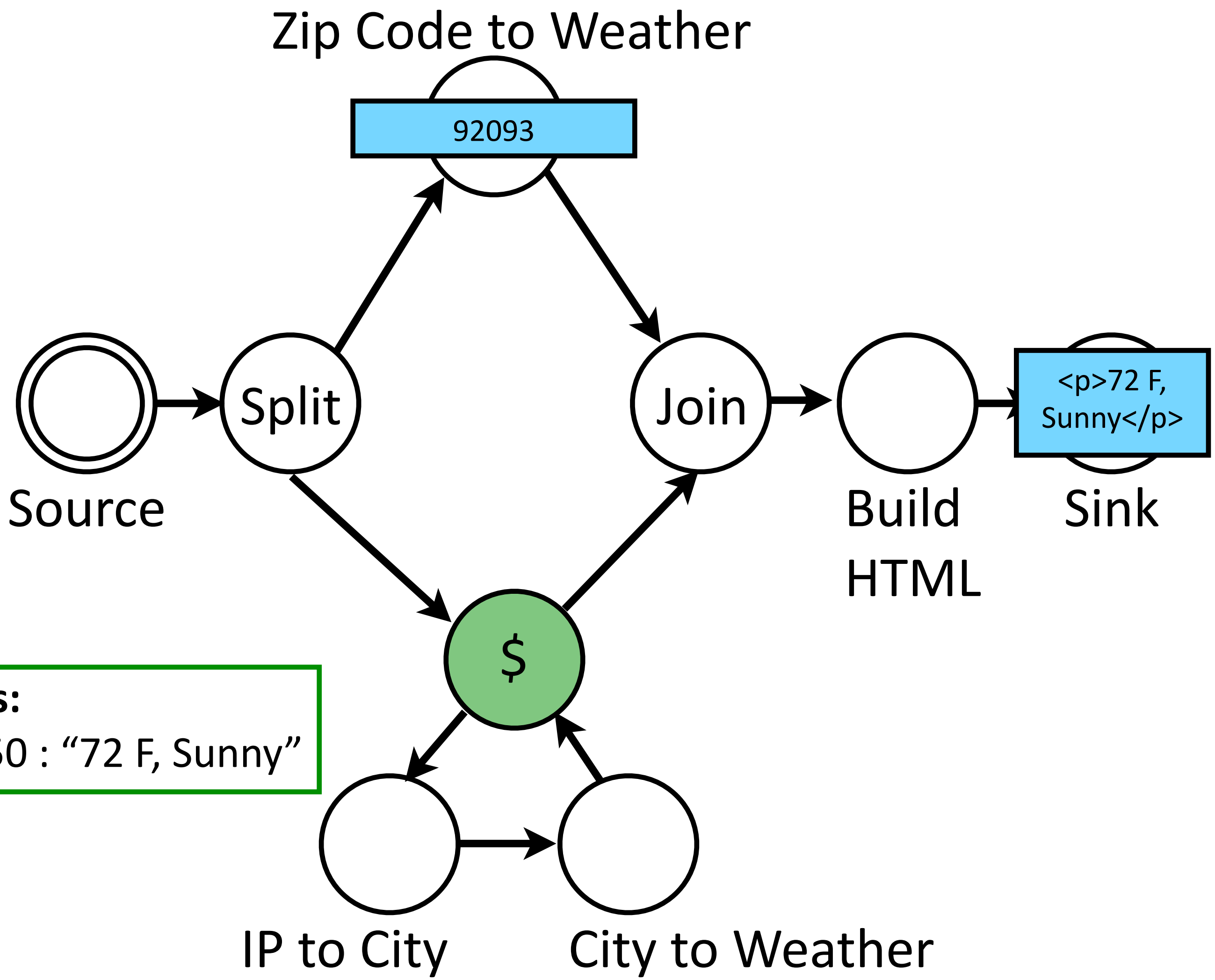


Cache Contents:
137.110.222.250 : "72 F, Sunny"

Caching {IP to City, City to Weather}

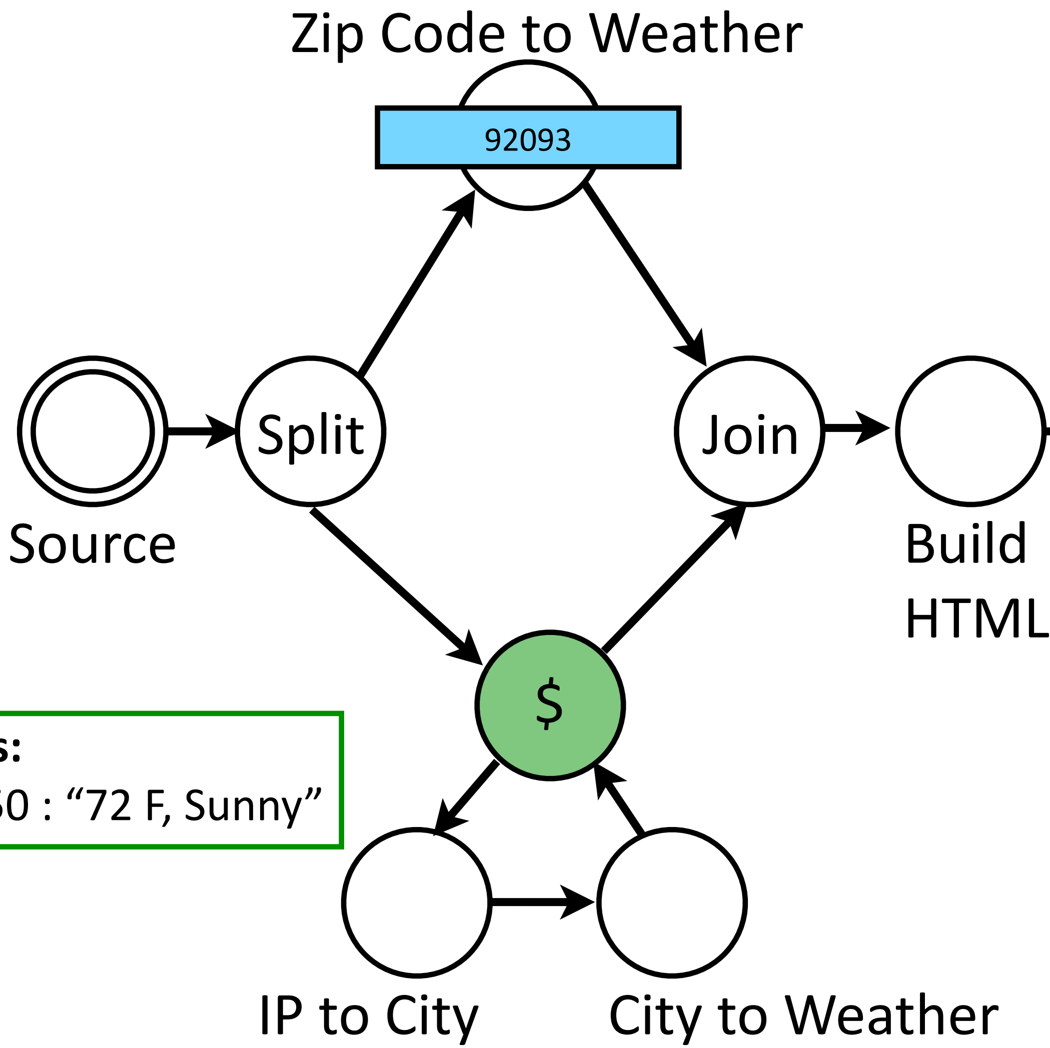


Caching {IP to City, City to Weather}



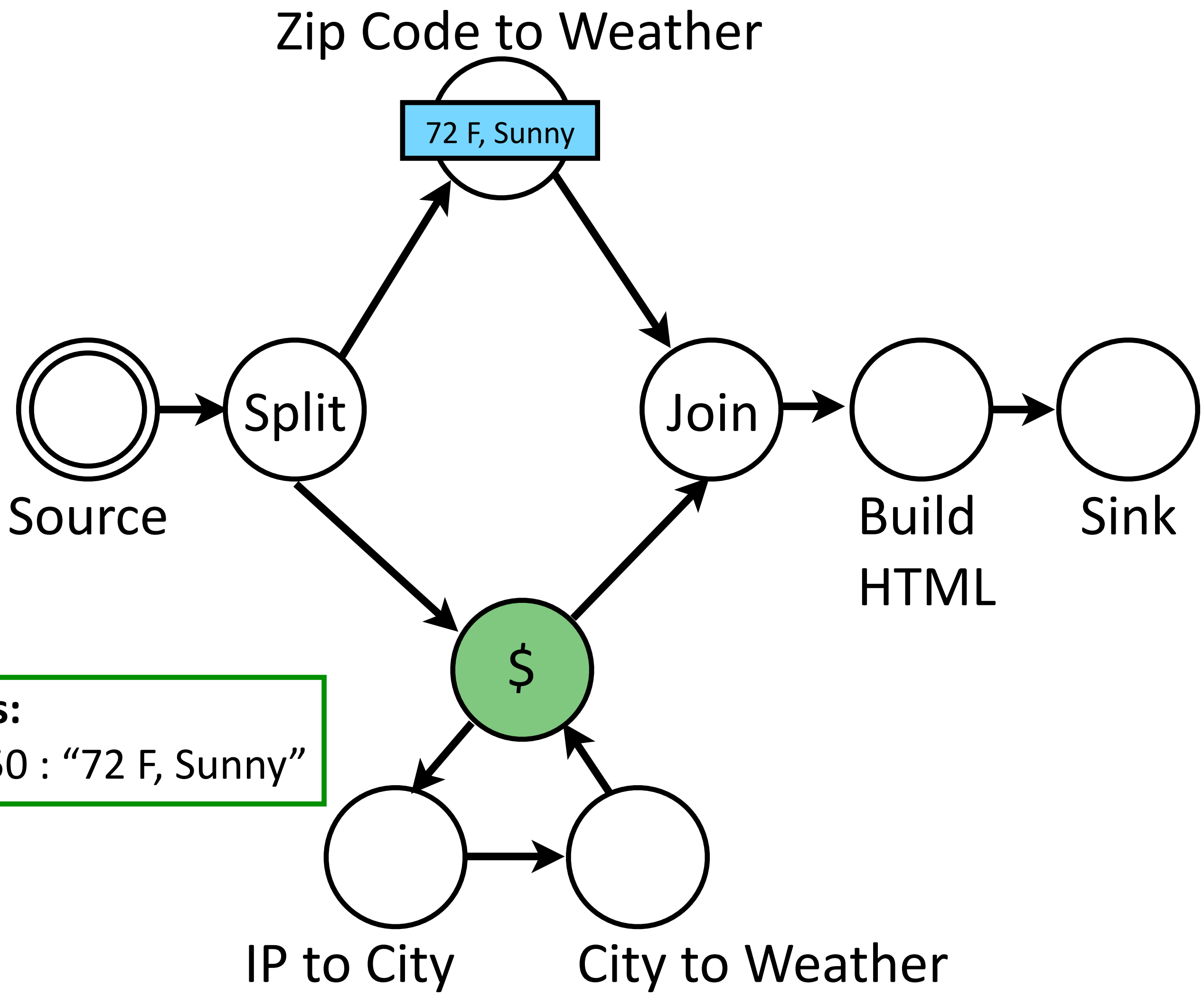
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



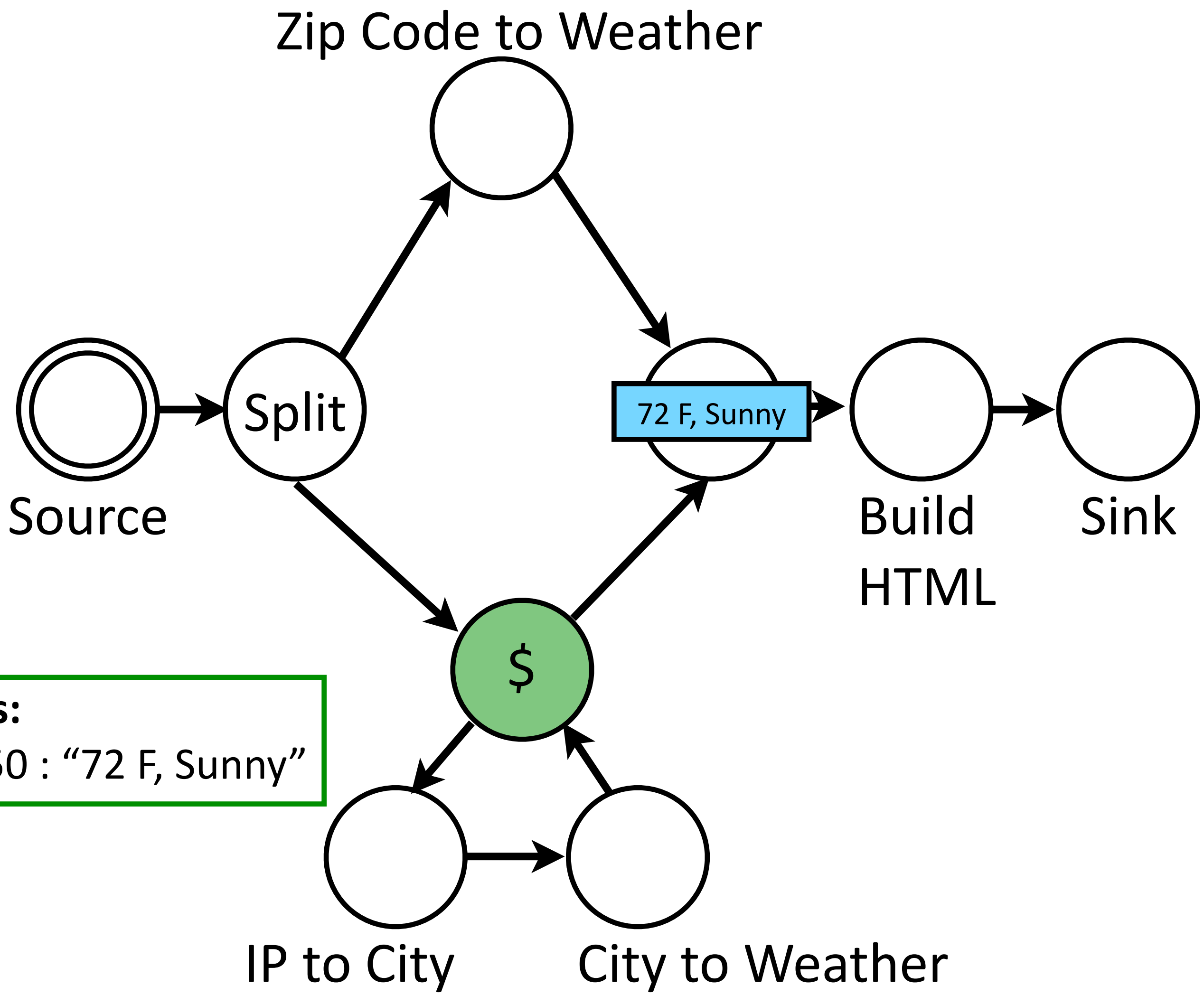
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



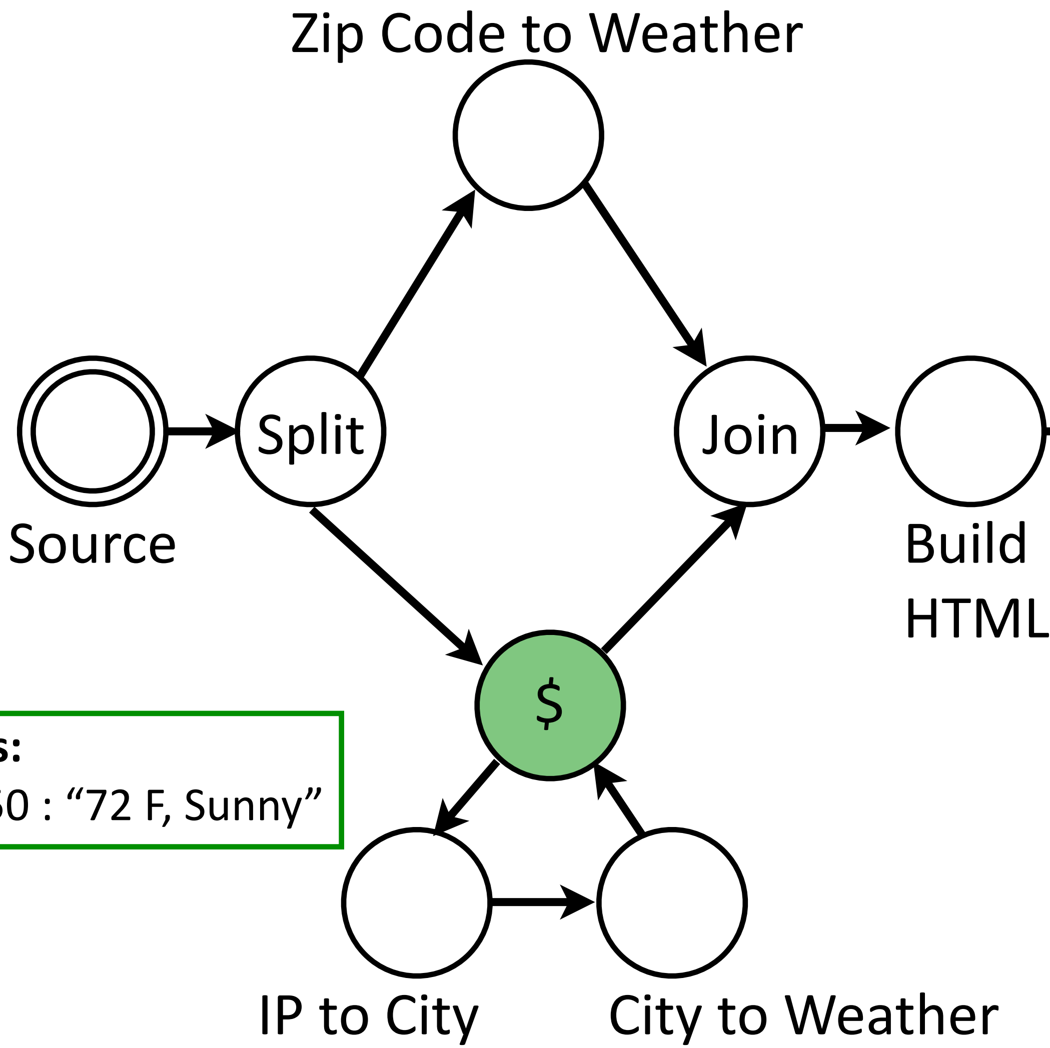
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



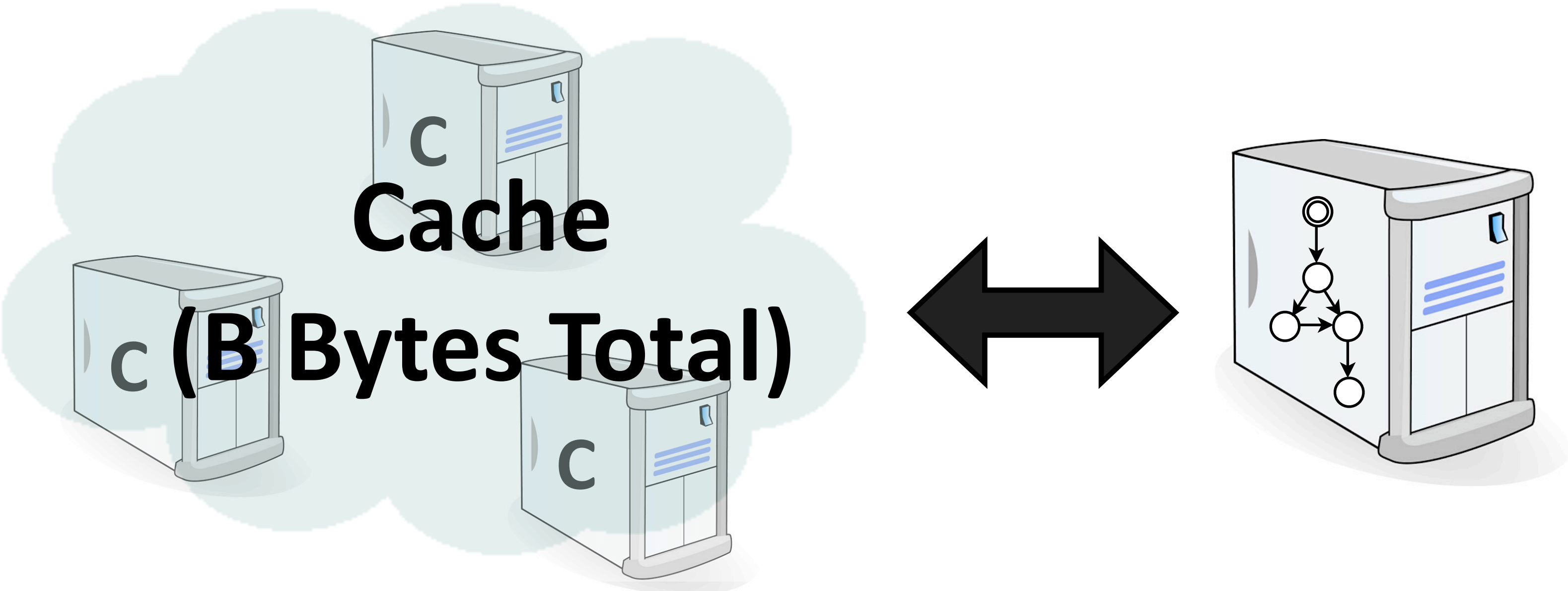
Cache Contents:
137.110.222.250 : “72 F, Sunny”

Caching {IP to City, City to Weather}



Cache Contents:
137.110.222.250 : “72 F, Sunny”

Simplifying Assumptions

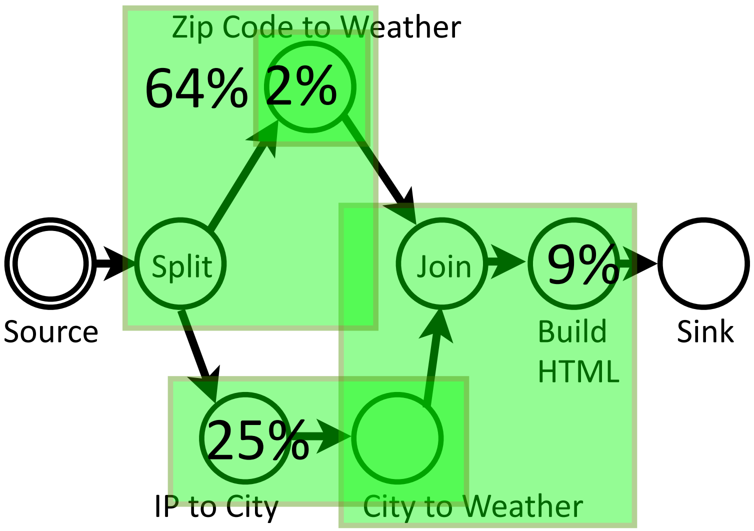


- Data center, single administrative domain
- Caching provided by cluster of *caching servers*
- Service runs on single machine, makes calls to external services during execution
- Goal: allocate B total bytes from cache servers to a service

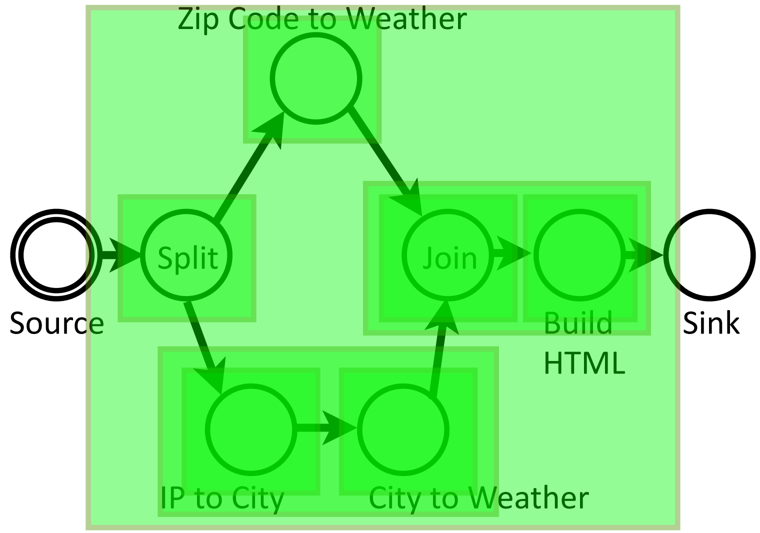
Fluxo Components

- Fluxo Runtime
  – Provides tracing and simulation functionality
  – Produces ordered stream of *events* as service runs
- Fluxo Optimizer
  – Takes stream of events and service graph, produces a *caching policy:* {<service subgraph, cache size> pairs}
  – Evaluates N random cache policies, hill-climbs from the top K policies
    - In our experiments, N=20,000 , K=200
  – To evaluate a policy, simulate its performance on recorded event stream

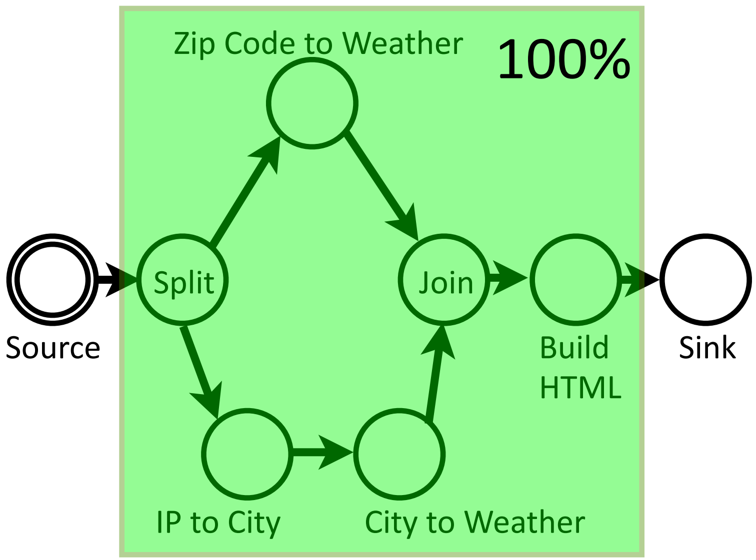
Evaluation - Reference Policies



Random
(sample)

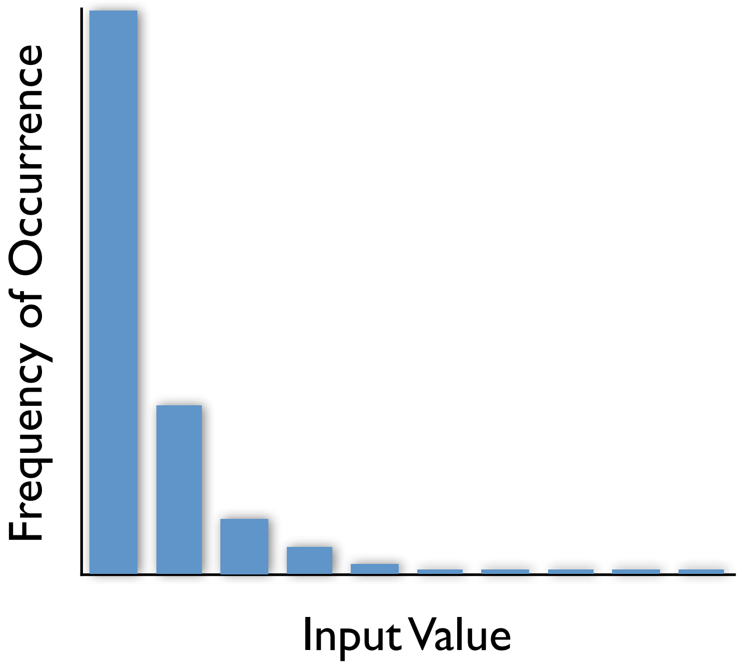


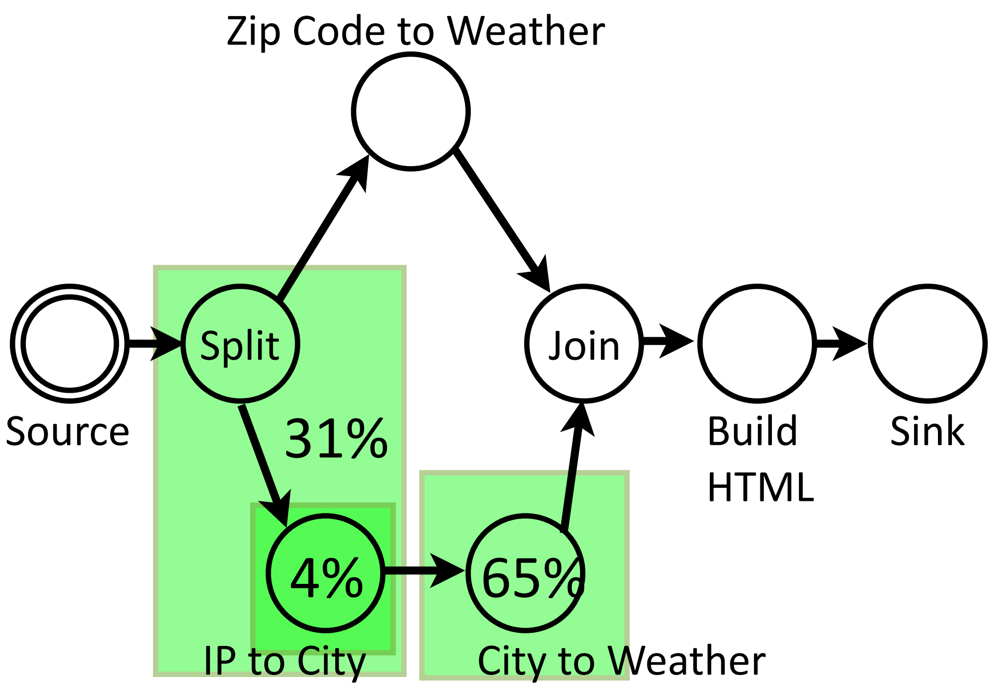
Uniform
(subset)



All-Encompassing

Results





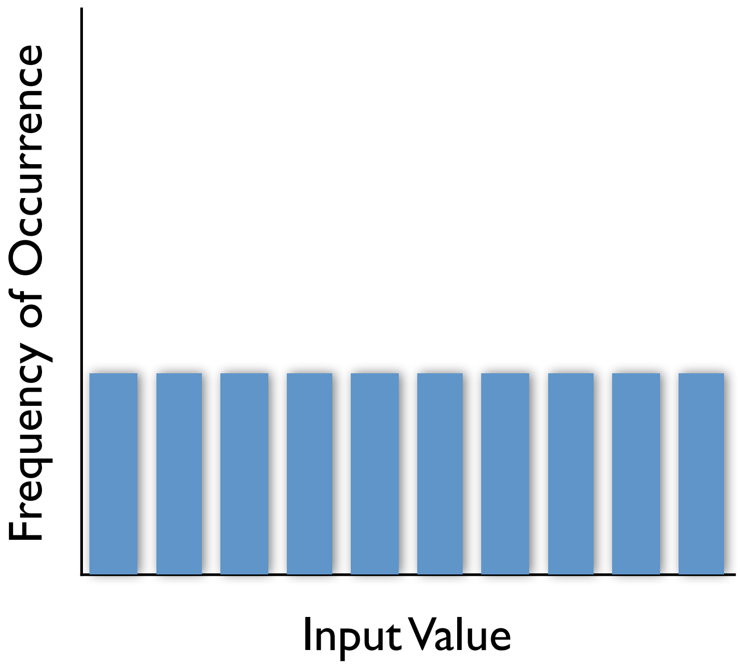
Median Latency Improvement:

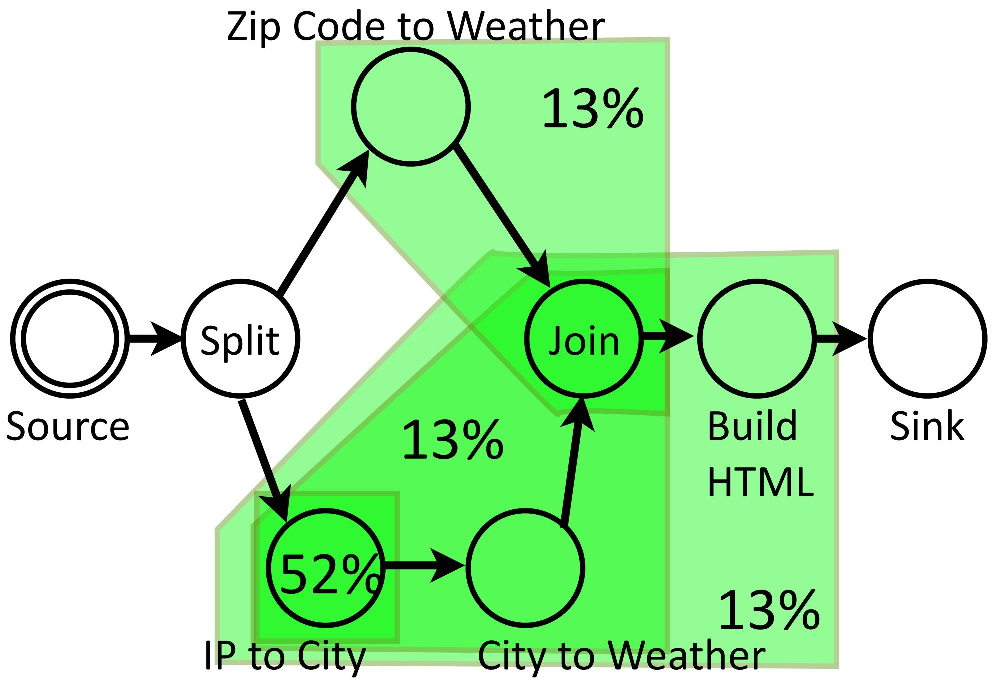
vs. Random: +5%

vs. Uniform: +6%

vs. All-Encompassing: +5%

Results





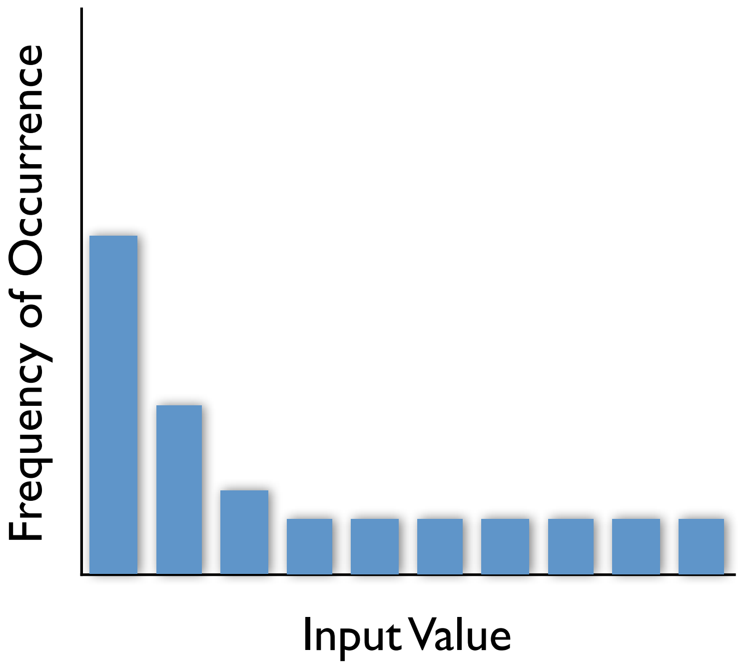
Median Latency Improvement:

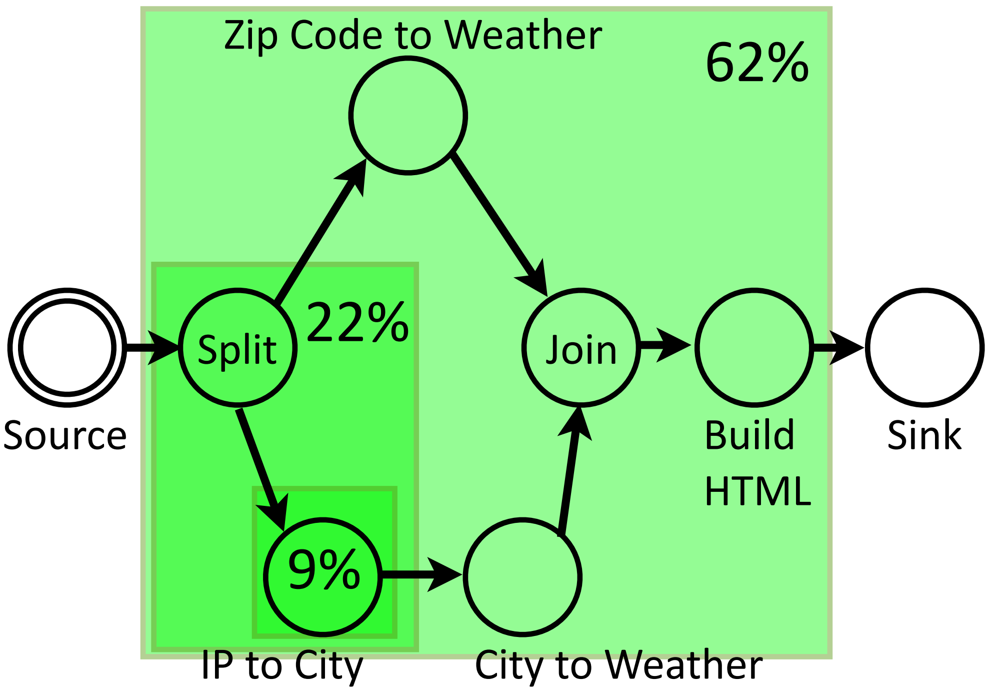
vs. Random: +12%

vs. Uniform: +15%

vs. All-Encompassing: +3%

Results





Median Latency Improvement:

vs. Random: +12%

vs. Uniform: +17%

vs. All-Encompassing: +1%

Future Work

- Evaluation on real service with real workload
- Scaling optimizer's analysis
  - Considering parallelized analysis, more aggressive result memoization, more sophisticated ML
- Seems hard to beat all-encompassing cache
  - Might be an artifact of test service
- Imperative programs?

Conclusion

- Fluxo:
  - Dataflow model of Internet services
  - Runtime tracing + model = caching policy
  - Simulation and search to converge on good policy

Thanks to John Wilkes for shepherding this work,
and to MSR for travel funding