A Platform for Matching Context in Real Time

Andrei Olaru, Adina Magda Florea

cs@andreiolaru.ro

University Politehnica of Bucharest

23.06.2015

- Context
- Formal Model
- Graph Matching
- Platform
- Experiments
- Conclusions

A Platform for Matching Context in Real Time

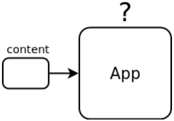
overview

Problem Context

This research is framed by the AmIciTy initiative, with the purpose of creating a software infrastructure for Ambient Intelligence (AmI) applications, that handles context at its constructive level.

[http://aimas.cs.pub.ro/amicity]

application receives message →



Questions asked:

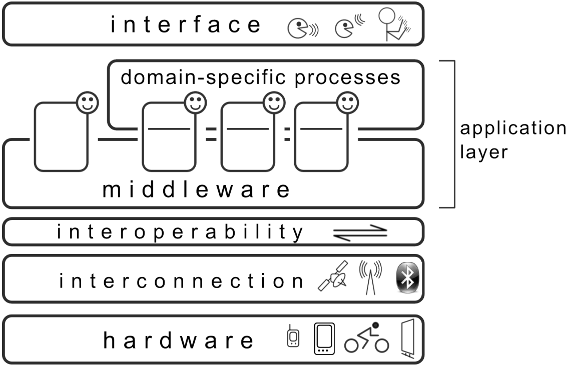
- How to represent message content as context information?
- Is content relevant to the activity of the application?
- How to integrate received content with current knowledge?
- What information should be sent to other applications / users?

How to integrate these processes in an[y] application?

Problem Context

Solution: a multi-agent system that handles context information accross the AmI ecosystem and provides it to applications → works as a context-ware middleware.

[Olaru et al., 2013]



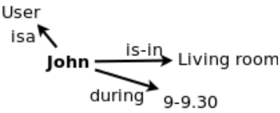
· What representation to use and how to work with it?

Formal Model

- We represent the information about the current situation as a Context Graph (directed graph with labeled nodes and optionally labeled edges), [Olaru et al., 2011]
- known situations as Context Patterns (graphs with some unlabeled nodes),
- and we use context matching (matching a pattern against a graph) to [Olaru et al., 2013]
  - detect whether new information is relevant
  - detect if we are in a known situation and potentially decide upon action to take
  - detect interests of other agents

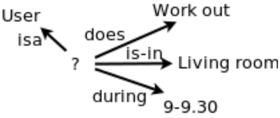
Formal Model

Current situation, as detected by smart bracelet:



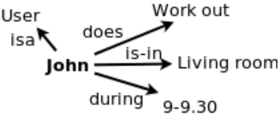
John is in the living room, during the interval $9 - 9^{30}$ (activity is unknown)

Known situation:



users work out in the living room during $9 - 9^{30}$

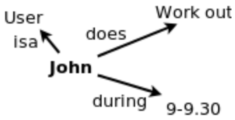
Inference:



John is probably working out

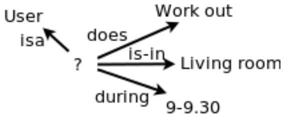
Formal Model

Current situation, as detected by smart bracelet:



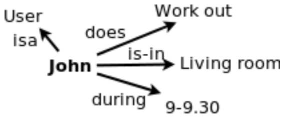
John is working out, during the interval $9 - 9^{30}$ (location is unknown)

Known situation:



users work out in the living room during $9 - 9^{30}$

Inference:



John is probably in the living room

Graph Matching Algorithms

· The problem of graph matching (MCS isomorphism) is NP-complete.

· Algorithms for graph matching normally work on unlabeled, sometimes undirected graphs.

Some examples include:

- Larossa – using CSP solving for exact matching [Larrosa and Valiente, 2002]
- Bron & Kerbosch – using maximal cliques in the modular product of the two graphs [Bron and Kerbosch, 1973]
- Koch – similar, but using the modular product of edge sets [Koch, 2001]

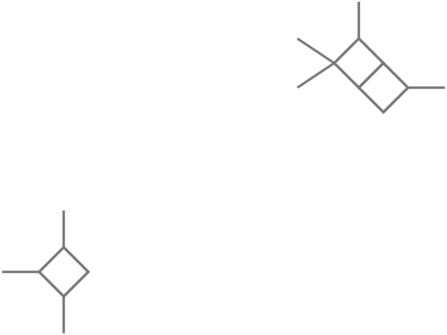
Graph Matching Algorithms

· We have developed the QuickMatcher algorithm in 2013, targeting specifically the problem of context matching. [Olaru, 2013]

- it starts from single-edge matches between the graphs and grows them to reach a maximal match.
- a match has a frontier, immediate merger candidates, and outer merger candidates.
- candidates need only be searched for once; when merging two matches, candidates can be computed through set operations.

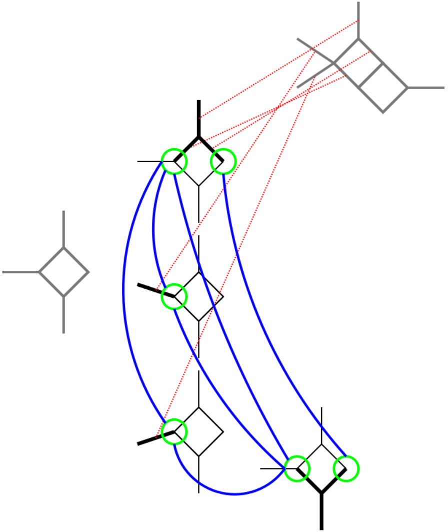
· The algorithm has outperformed classic matching algorithms, after they have been adapted to the context matching problem. [Dobrescu and Olaru, 2013]

Graph Matching Algorithms



graph & pattern

Graph Matching Algorithms



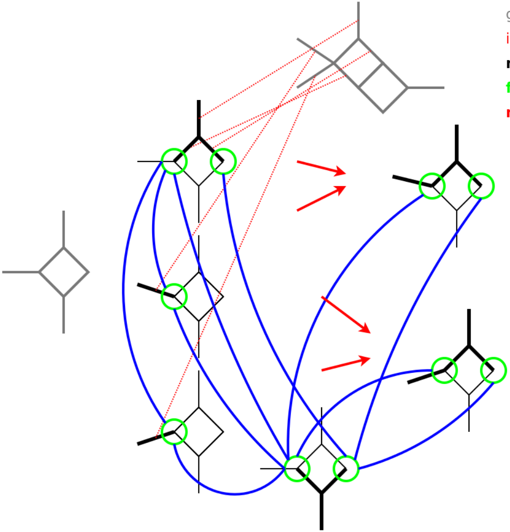
graph & pattern

isomorphism

matched part

frontier

Graph Matching Algorithms



graph & pattern

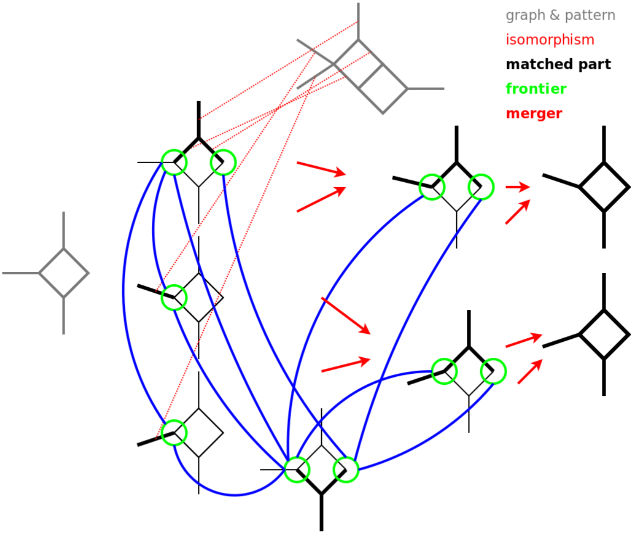
isomorphism

matched part

frontier

merger

Graph Matching Algorithms



graph & pattern

isomorphism

matched part

frontier

merger

CCM Platform

· the Continuous Context Matching Platform

Objective: create a platform for context matching that can be used by an agent that uses one context graph and multiple context patterns

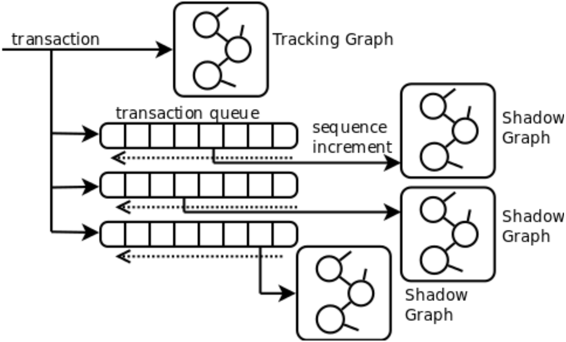
· The context graph changes incrementally, through the addition and removal of edges and nodes.

Challenges:

- don't miss changes while performing the matching
- keep information about the parts of the graph that don't change
- run in the background

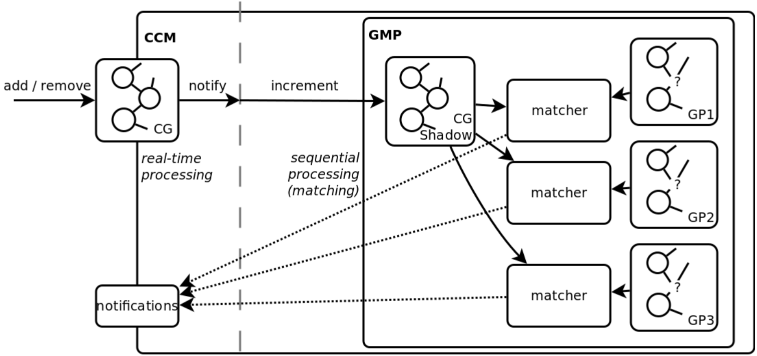
CCM Platform

- transactions as atomic sets of operations (add/remove node/edge)
- a TrackingGraph stores a queue of transactions applied to it
- ShadowGraphs are matched against the patterns, after each transaction is applied



CCM Platform

- the matching process runs in the background
- a shadow graph is used: after each transaction is applied, matching is incremented for each pattern
- notifications are produced when a match with certain parameters is found



CCM Platform

- partial matches are stored, complete with their data → no need to recreate them each time, since patterns modify rarely and the Context Graph is modified incrementally.
- when an edge is removed from the CG → matches containing it are removed.
- when an edge is added to the CG → the new single-edge match (if any) is checked against matches containing neighbor edges.

CCM Platform

· the Continuous Context Matching Platform uses a (tracking) graph and a set of patterns and allows the user to:

- receive a notification whenever a specific pattern is matched.

```
addMatchNotificationTarget(ContextPattern pattern, MatchNotificationReceiver receiver);
```

- receive a notification whenever a match with less than a specific number of missing edges (k threshold) is found.

```
addMatchNotificationTarget(int thresholdK, MatchNotificationReceiver receiver);
```

- start / stop the background matching process.

```
startContinuousMatching(); stopContinuousMatching();
```

- start a persistent background matching process on a specified graph (against the known set of patterns) and receive notifications whenever matches are found.

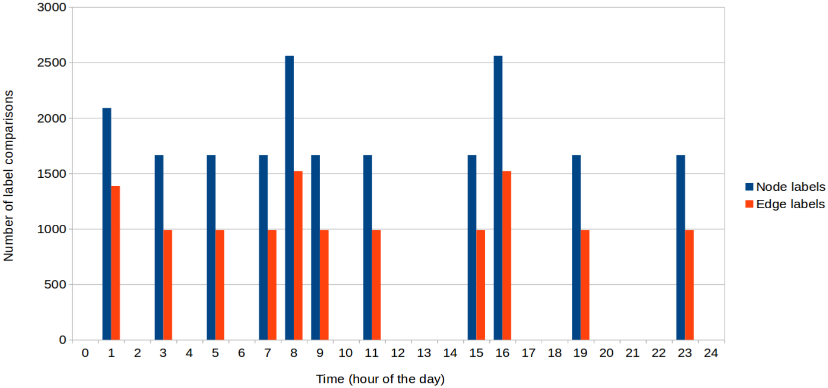
```
startMatchingAgainstAllPatterns(Graph graph, int k, MatchNotificationReceiver receiver);
```

- start a persistent background matching process of the context graph against a specific pattern.

```
startMatchingAgainstGraph(Graph pattern, int k, MatchNotificationReceiver receiver);
```

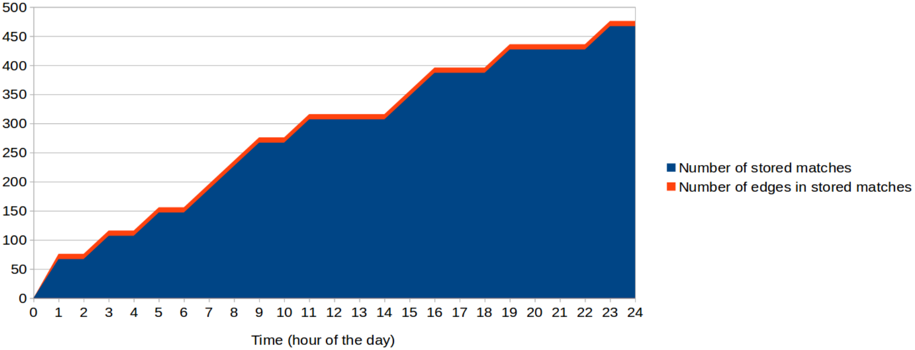
Experimental Results

- experimental setup: 24h-long synthetic scenarios



Label comparisons

Experimental Results



Matches stored in memory

- most matches are single-edge matches.
- in the future, a mechanism will be developed to select which single-edge matches to keep.

Conclusion and Future Work

Done

- built a context-matching platform for matching multiple context patterns against the same context graph.
- the platform can be added as a component that works in the background, tracking all changes and notifying the host applications about matches.

Future work

- optimize memory usage to store less single-edge matches.
- learn patterns of activity → use publicly available datasets.
- create large scenarios with a large number of agents, study performance.
- use the CCM Platform as a component in every agent in an AmI-oriented MAS.

Thank You!

Any Questions?

cs@andreiolaru.ro

Bron, C. and Kerbosch, J. (1973).

Algorithm 457: finding all cliques of an undirected graph.
Communications of the ACM, 16(9):575–577.

Dobrescu, A. and Olaru, A. (2013).

Graph matching for context recognition.
In Dumitrache, I., Florea, A. M., and Pop, F., editors, Proceedings of CSCS 19, the 19th International Conference on Control Systems and Computer Science, May 29-13, Bucharest, Romania, pages 479–486. IEEE Xplore.

Koch, I. (2001).

Enumerating all connected maximal common subgraphs in two graphs.
Theoretical Computer Science, 250(1):1–30.

Larrosa, J. and Valiente, G. (2002).

Constraint satisfaction algorithms for graph pattern matching.
Mathematical structures in computer science, 12(4):403–422.

Olaru, A. (2013).

Context matching for ambient intelligence applications.
In Björner, N., Negru, V., Ida, T., Jebelean, T., Petcu, D., Watt, S., and Zaharie, D., editors, Proceedings of SYNASC 2013, 15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, September 23-26, Timisoara, Romania, pages 265–272. IEEE CPS.

Olaru, A., Florea, A. M., and El Fallah Seghrouchni, A. (2011).

Graphs and patterns for context-awareness.
In Novais, P., Preuveneers, D., and Corchado, J., editors, Ambient Intelligence - Software and Applications, 2nd International Symposium on Ambient Intelligence (ISAmI 2011), University of Salamanca (Spain) 6-8th April, 2011, volume 92 of Advances in Intelligent and Soft Computing, pages 165–172. Springer Berlin / Heidelberg.

Olaru, A., Florea, A. M., and El Fallah Seghrouchni, A. (2013).
A context-aware multi-agent system as a middleware for ambient intelligence.
Mobile Networks and Applications, 18(3):429–443.

Thank You!

Any Questions?

cs@andreiolaru.ro