Traces in the Analysis of Heterogenous (Numeric) Data (in SNs)

Markus Scheidgen

Department of Computer Science, Humboldt Universität zu Berlin
Unter den Linden 6, 10099 Berlin, Germany
{scheidge}@informatik.hu-berlin.de

Abstract. Markus: TODO

1 Introduction

1.1 The examples

All examples in this paper are taken form the domains *wireless mesh networks* (WMN) and *wireless sensor networks* (WSN). Specifically, all example are taken from the *Humboldt Wireless Lab* (HWL) test-bed. The HWL test-bed is a WSN based on WMN technology. Therefore, the HWL test-bed is both a WMN and WNS.

1.2 The nature of data and information

Terms like data or information are ambiguous and hence have potentially different and confusing means is different communities. *Disclaimer:* The following is not an attempt to provide some sort of commonly accepted information theory; it only serves the cause of this paper.

We start with the smallest pieces: *atomic pieces of data* (APD). APDs do not contain other data but themselves. An APD has an *identity*, *value*, and *data type*. The set of values of all APDs of a certain type form the *defining set* of that data type. Many data types are defined through (subsets) of real or natural numbers, such as temperature values between -30 and 100 degree Celsius. A set of labels is also a typical APD type. But data types are not limited to numbers and strings. Tuples of numbers, bitmaps, graphs, etc can also be the values of APDs, but only if these APDs are never considered to consist of multiple parts. Each APD has an identity. Two APDs from type integer, are not necessarily the same just because their values are equals.

Because each single APD does not have any context, it is considered *data* and not information. But APDs can be linked to each other. We consider graphs of APDs as *information*. The vertices are takes from a set of APDs and edges connect different APDs. In such a graph each APD has a *context*: that are the neighbors of an APD.

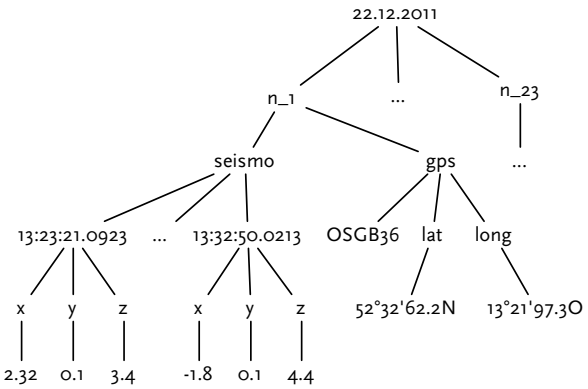


Fig. 1. Experiment data (information) from the HWL a WMN/WSN with seismo and gps sensors

We assume that each information graph contains a designated information tree. An *information tree* is a directed and acyclic graph subgraph that covers all vertices (i.e. all APDs). Edges are directed from *child* to *parent*. One child can have multiple parents, one parent can have multiple parents, diamonds are allowed. An information graph can contain multiple information trees, but one tree must be designated and is called *the information (or composite) tree of an information graph*.

In an information tree edges provide an antisymmetric relationship between APDs. The reflexive transitive hull of this relationship induces a partial order among APDs in an information graph. We simple say an APD is *related* to another APD, iff these APDs are related with respect to this reflexive transitive hull. Note that children are only related to parents and not vice versa. The set of all related APDs of an APD is called *ancestors* of that APD. The joint of ancestors of a set of APDs is called the *common ancestors* of that set. The set of smallest ancestors is called the *closest ancestors* of a set of APDs. Two APDs are *siblings* if they have a non empty set of common ancestors.

The information graph if Fig. 1 for example comprises of all the data from an experiment with a WSN of 23 nodes and two sensors (seismo and gps). This includes APDs with several types: date, time, node identifiers, sensor identifiers, spatial axis, accelerometer readings, degrees (long/lat), and coordinate reference systems. The ancestors of each accelerometer reading contain a axis, a time, a node, and an experiment date. Each accelerometer reading and each gps coordinate at least share the experiment date as common ancestor. An accelerometer reading and a gps coordinate of the same node share also the node as common ancestor.

Information has also a type (*information type*). Consider the WSN from the last example: all possible information graphs from experiments with this WSN share a unique structure. There are constraints for the set of possible APDs and possible edges.

There are several technical systems to create, store, and access information. XML files can contain information trees (and graphs) where each entity (or attribute) represents an APD. XML schema define types of information. Relational databases organize information in tables and references between entries of different tables. Each value (specific entry, specific column) in a relational database table is a APD. Entries and relationships form links between APDs. Entity relationship diagrams or database schemas define types of information. Models (as in OMG) are information, objects (and attributes) are APDs. Meta-models define types of information. Within programming languages data structures (classes, structs, union, arrays and primitive types) are used to represent information. Further systems are based on ontologies or RDF.

There are several abstractions for the representation of information. We may call graphs of APDs information, but these graphs are actually only one possible abstraction. Other (somewhat limited) representations are list, maps, functions, terms, different forms of trees and graphs, algebras, vectors, matrices, tables, etc. Of cause different representations can be combined.

1.3 Analysis and Traces

The information graphs considered in this paper (e.g. the information graphs defined and exemplified in section 1.2) are to be analyzed. Information is usually created during experiments. These experiment results are on low level of abstraction and we want to extract information on a higher layer of abstraction. The term *analysis* describes the process in which new information graphs are created from existing information graph. An analysis can be divided into analysis steps. In each *analysis step* a distinct information graph is created. An analysis can have multiple steps and hence produce intermediate information graphs, i.e. intermediate results. Fig. 2 shows the information graphs of an example analysis.

When a new information graph is created during an analysis step, APDs of the new graph can be assigned to APDs of existing information graphs: new APDs are either directly taken (copy) or are computed from one or more existing APDs. This creates a directed relationship between APDs of different information graphs. This relationship is antisymmetric. The reflexive transitive hull links one APD to all those APD that participate in its creation (including the APD itself). We denote this relation *trance relation*. The trace relation induces a partial order. The ordered set of APDs that participate in the creation of an APD (including the APD itself) is called the *trace* of that APD.

Original information graph, intermediate and final results of an analysis as well as the trace relationship can be used to form a union information graph. The trace relationship and all information trees compose the composition tree of that new information graph. We call this information graph the *analysis information graph* and the original information graph the *experiment information graph*.

Analysis information graphs can be further extended. Depending on the type and semantic of APDs, we can define (usually equivalence) relationships on the values of APDs. For example: to denote close proximity GPS coordinates can be designated equivalent, timestamps can be designated equivalent if they shell

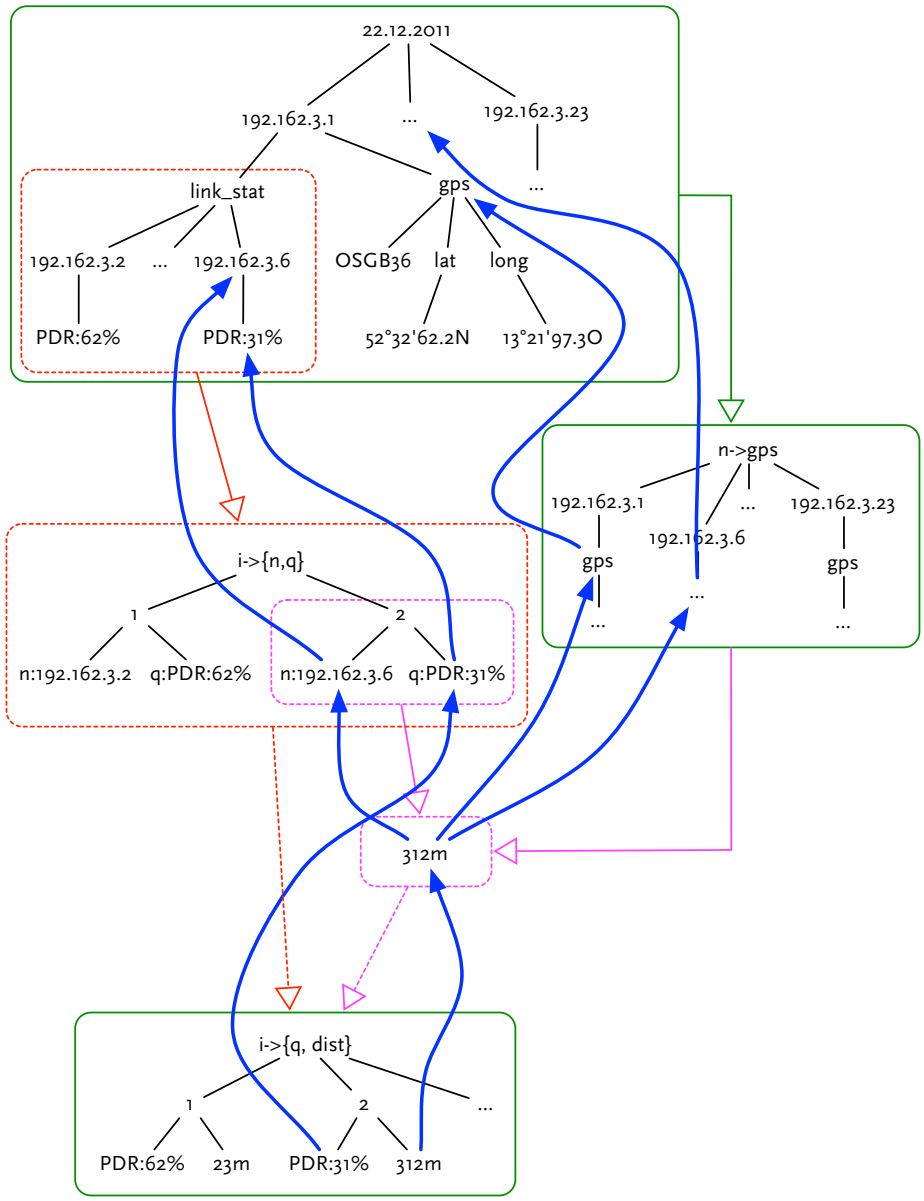
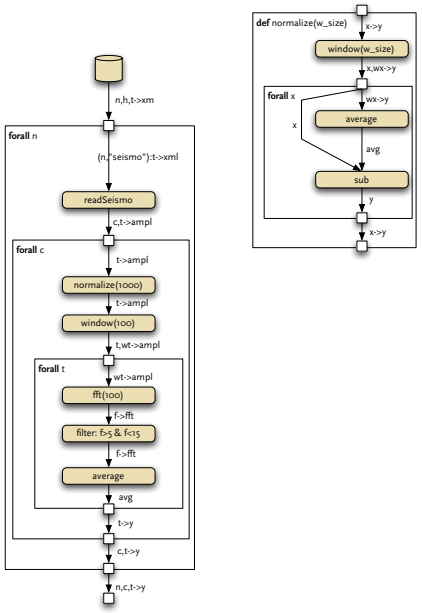


Fig. 2. An example for Information in different analysis steps and traces between its APDs. The example is taken from an experiment with the HWL network. The information obtained within the network includes information about links and link quality (packed delivery rate, PDR) and the position of the nodes (via global positioning system, GPS). The colored boxes mark distinct information graphs. The thin colored arrows mark analysis steps, the arrow point marks the created information graph. Green boxes mark information graphs that cover the whole experiment, other colors mark information graphs that only cover single samples. In the example, the red box describe the links of one network node, the purple boxes describe a single link. The thick blue arrows mark traces between APDs: the arrows show which APD is based on what other APDs.

denote the same point in time, etc. This way, two APDs can be *weak siblings* if they share ancestors with different identity but (similar or) same meaning.



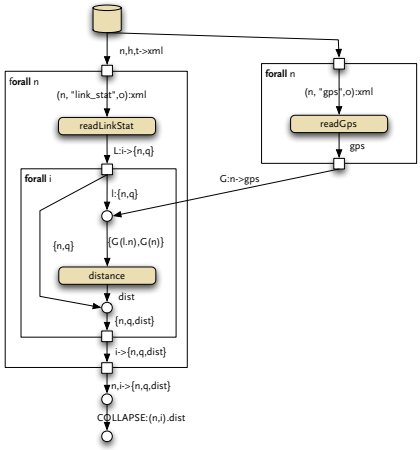


Fig. 4. Link distances composition

Fig. 3. Seismo analysis composition.

Examples taken from experiments with wireless mesh networks, the experiment pictured in Fig. 2. The gps coordinates of two wireless nodes are siblings with the experiment time as closest common ancestor. The gps coordinates of two wireless nodes that form a link are siblings to the link quality of that link with the link as closest common ancestor; further ancestors are the nodes of the link and the experiment time.

Fig. 3 and Fig. 4 examples for the analysis of HWL ClickWatch data. These activity diagrams choreograph analysis steps (activities). Both are supposed to be performed on HWL ClickWatch data. This is are information graphs (trees). In those nodes have handlers and handlers have XML values at different points in time: hence n, h, t, xml as starting information graph. XML values are part of the information graph.

1.4 Applications

The described concepts of APD, information and analysis are not new and performed in many fields. What is somewhat new is the notion of traces. In complex analysis scenarios traces allow to find siblings and common ancestors that allow to put the results and intermediate results of an analysis in relation.

In experiment scenarios, where heterogenous data is produced and analyzed, different types of data is analyzed in different analysis steps, threads of analysis steps aggregate the different types of data with each other, and with each further analysis step the created results represent information on increasingly higher levels of abstraction.

In such scenarios, it is not trivial to identify all ancestors and siblings of a resulting APD. Heterogenous data analysis produces different results. During interpretation of these results, the APDs of the different results must be related to each other.

An example from WMN/WSN. In WMN/SN experiment scenarios we have a large number of similar (or identical) data sources: the nodes of the network. But each node produces a large number of different APDs. Different sensors, protocol entities, etc. produce all different data. Analysis in this context means that for the different data types on each of the identical nodes different threads of analysis steps are applied. The results are many and complex information graphs that cover different aspects of the network. Trace based knowledge about ancestors and siblings becomes key.

Why is it key to identify siblings? The seismo sensors in 100 node network produce time signals: one might want to compare the time signals of nodes that are geographically close to each other. Close geographic positions are common ancestors for the time signals. The nodes form links (knowledge about links is a analysis result): it is important to compare the characteristic (analysis result) of one node to the characteristic of linked nodes. Here, nodes are ancestors of the characteristics; the nodes in a link have common ancestors (the links). Looking at a the characteristics of a link or a route (connected links), it is important to identify the characteristics of all nodes in the link or the route. Looking at the time signal of one sensor, one might want to see the time signals of other sensors of that node, or nodes that have links to the node, or nodes in close proximity. Looking at a sensor reading, one might want to see node characteristics recorded at a point in time close to the time of the sensor reading. Etc. etc. etc.

2 A Framework for Analysis with Traces

2.1 Requirements

– Large number of data. Information graphs must be scalable. This means that APDs must be indexed somehow to access distance subgraphs efficiently. This is also true for final and intermediate results.

3 Related Work

4 Conclusions