Towards Using Location Poly-Hierarchies for Energy-Efficient Continuous Location Determination

Maximilian Schirmer and Hagen Höpfner
Bauhaus-Universität Weimar
Media Department / Mobile Media Group
Bauhausstraße 11, 99423 Weimar, Germany
maximilian.schirmer@uni-weimar.de, hoepfner@acm.org

Keywords

Location Determination, Location Poly-Hierarchies, Energy Efficiency

ABSTRACT

Location awareness is a key feature of mobile information systems. Typically, location is determined by interpreting a set of measured positions. Various approaches for position determination do exist. They vary greatly in their precision, applicability, and energy requirements. As mobile devices are battery-driven, energy is one of the most limiting factors for the system's uptime. Locations have a hierarchical nature and location-based applications differ in their required precision. In this paper, we present three approaches that utilise location poly-hierarchies in order to reduce the energy demand of continuous location determination: (1) We analyse the dependencies among the different hierarchy levels, (2) we incorporate an adaptive delay between measurements based on the hierarchy level and the calculated minimal required time to change, and (3) we select appropriate positioning techniques for each hierarchy level. We implemented and evaluated our approaches.

1. INTRODUCTION AND MOTIVATION

Navigation systems, social network apps, tourist information systems, event information systems, shop finders and many more heavily rely on position data of their users. Consequently, almost all modern smartphones supply positioning techniques. The most popular positioning technique is the Global Positioning System (GPS). However, even devices that do not provide GPS hardware are able to locate themselves using alternative techniques such as geotagged Wi-Fi hotspot and cell tower databases (db), wireless signal triangulation/lateration, or geotagging. Although all of them allow localisation of a mobile device, they vary dramatically in precision, applicability, hardware requirements, and energy demands. A GPS request requires a GPS receiver with a much higher energy demand compared to an on-device database lookup that is based on already localised cell towers or Wi-Fi hotspots [4]. However, in an outdoor scenario, GPS is far more precise than the analysis of location information of connected Wi-Fi hotspots or cell towers [22]. On the other hand, indoor GPS positioning is almost impossible because of the occlusion and shielding of satellite signals created by building structures.

Mobile devices are battery-driven. So, energy is one of the most limiting factors for their uptime. Additionally, the user acceptance of location-based or context-aware mobile applications is negatively influenced when the applications heavily strain the mobile devices' batteries. Furthermore, location-based applications differ in their required precision [16]. While the name of the city a user or a device is currently located in might be appropriate for an event information system, a navigation system might demand for exact coordinates or street names at least. A common representation of locations follows their hierarchical nature. In a hierarchical model, earth is divided into continents, continents into countries, countries into states, states into cities, cities into streets and streets into street numbers, and so on. Consequently, determining low-level location information in this hierarchy also determines upper levels and contains information that is connected to these levels. In addition to this, locations only change if the device is moving. While GPS coordinates might change with each movement, city information is stable until the device leaves the city. Hence, the system might wait with the next energy-demanding location determination on the city level until the device possibly left the city. In this paper, we present three approaches that utilise location poly-hierarchies for reducing the energy demand of continuous location determination:

1. We analyse dependencies among different hierarchy levels.
2. We postpone position measurements based on a calculated minimal time that is required for leaving the current location.
3. We select appropriate positioning techniques for each hierarchy level.

Our preliminary experimental results show that there is a strong potential in these techniques to reduce the energy demand compared to continuous GPS polling.

The remainder of the paper is structured as follows: Section 2 discusses related work. Section 3 introduces the concept of location hierarchies. Section 4 presents the three aforementioned approaches. Section 5 describes the evaluation approach and the results. Section 6 summarises the paper.

2. RELATED WORK

Our work mainly overlaps with the research fields *location models* and *energy-aware computing*. Location models form the basis for all high-level operations on location data. Consequently, the frequent and enduring use of location data in mobile computing immediately raises the issue of the mobile devices' limited energy

24^{th} GI-Workshop on Foundations of Databases (Grundlagen von Datenbanken), 29.05.2012 - 01.06.2012, Lübbenau, Germany.

resources. The field of energy-aware computing presents a variety of concepts and methods to compensate for these constraints.

2.1 Location Models

Location models as core components of location-based applications represent location information and spatial (or even spatio-temporal [15]) relationships in data. They help to express relative locations, proximity, and allow users to determine containment of locations or connectedness of relationships.

The authors of [21] present in great detail the broad variety of location models that have been developed in recent years of active research. A key factor for distinguishing and characterising location models is their way of representing spatial relationships. According to [1], they can be categorised into set-based, hierarchical, and graph-based models. Hybrid models that combine several aspects exist as well. Figure 1 presents an overview of the three main concepts. In the illustrated examples, the set-based approach is the least expressive one, as it only models the fact that there are two distinct locations within a set of locations, and a set of coordinates is assigned to each location. The hierarchical model adds containment information, and the graph-based model adds connectedness as well as distance in the form of edge weights.

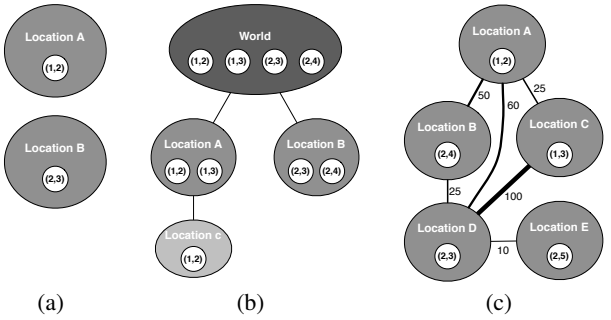


Figure 1: Examples for different popular location models: set-based (a), hierarchical (b), and graph-based (c). The edge weights in the graph-based model represent distance information.

Hierarchical location models as a special case of set-based models represent containment relationships between different levels of the model and are widely used as basis for location-based applications [14, 2, 6]. They cannot represent distance information or directly encode proximity, but they have great advantages in traversal and for containment queries. They are very close to the common human understanding of locations. Almost everyone understands the widely acknowledged segmentation of locations into administrative regions (country, state, city, street, etc.). At this, on a city level, a lot of implicit information can be derived through the top-level relationship to a state or country (e.g., administrative language, local cuisine, prevalent religions).

2.2 Energy-aware Computing

Energy-aware computing recognises the need for energy as a factor in modelling and implementing computing systems in order to manage and reduce their energy demand. This includes both hardware and software systems. While energy-aware hardware has been under active research for many years, energy-aware software is still a novel and underestimated field of research. Hardware solutions such as sleep modes or performance scaling cannot be directly transferred or adapted to software systems. Energy-aware software requires dedicated software engineering with new concepts and algorithms [9].

One concept of energy-aware computing is *resource substitution* [3]. It is based on the observation that in most cases, alternative resources exist for a given resource. These alternatives often vary greatly in their costs (e.g., computing power, storage capacity, or energy demands), but also in their accuracy, granularity, and frequency of data updates. This directly influences their appropriateness for substitution. In general, resource substitution favours resources with a lower cost (energy requirements) over expensive (high-energy) alternatives. In many cases, a high-energy resource is not necessarily required and can be substituted without measurable impact on system performance or user acceptance [19].

The authors of [17] utilise resource substitution in the form of *sensor substitution*. In a location-based context, data from a GPS device is often substituted with triangulation or lateration data from cell towers or Wi-Fi stations. A comparable concept is *sensor triggering*, where logical dependencies between different sensors are used. When low-energy sensors detect changes in the environment, a detailed update with high-energy sensors is triggered. An experiment described in [17] shows that low-energy accelerometer data can be used to trigger high-energy GPS sampling. This approach greatly reduces the energy demand of location determination for mobile applications that do not require a gap-less reconstruction of routes. This triggering approach has also been applied in the area of civil engineering, where it is critical that autonomous sensor nodes in buildings gather highly detailed data when vibrations occur. In the "Lucid Dreaming" system [13], a low-energy analogue circuit is sufficient to watch for these environment changes. It triggers a high-energy microcontroller-based sensor to gather the required fine-grained data.

3. TERMS AND DEFINITIONS

According to [18], "*location of an object or a person is its geographical position on the earth with respect to a reference point.*" From our viewpoint, this definition is too restrictive, as geographic positions are points. In contrast to a point, a location has a spatial extent. Due to [5], "*geographic location is the text description of an area in a special confine on the earth's surface.*". However, an *area* is a set of geographical positions. So, we use a set-oriented definition: *A location is a named set of geographical positions on earth with respect to a reference point.* In a two-dimensional coordinate system, e.g., the location of a building is given as a set, where each position (point) belongs to the building's area. We do not discuss the calculation of point sets here, but rather refer to techniques of geographical information systems (GIS) [7]. The location models presented in Section 2.1 describe relationships among locations. However, reality requires a more sophisticated location model. Istanbul, as the capital of Turkey, belongs to Europe and Asia. The same issue holds for Russia, which is located in Europe and in Asia, too. Another problem results from enclaves: Kaliningrad is part of Russia, but this information is not sufficient to decide whether Kaliningrad belongs to Europe or Asia. The solution for these problems is to use set overlaps instead of containment relationships in combination with a poly-hierarchical location model [11].

Figure 2 illustrates the simplified poly-hierarchies for Istanbul and Kaliningrad, represented as directed acyclic graphs. Each node is a location and each directed edge represents that the child node belongs (semantically) to the parent node(s). Moreover, the location poly-hierarchy LPH has a unique root node because the entire coordinate system is closed in case of locations (all considerable positions are elements of the set of all positions on earth). We do not discuss the construction of LPH in this paper, but assume that an

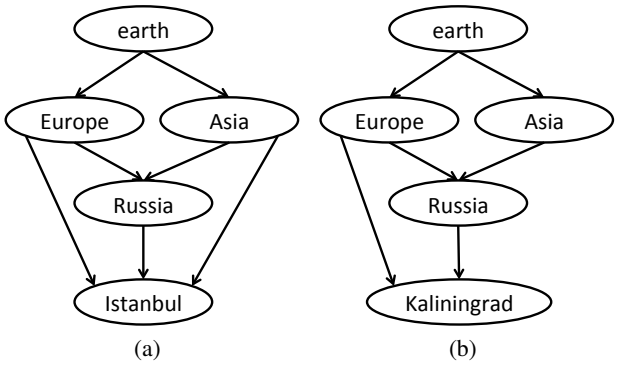


Figure 2: Poly-hierarchical example locations: Istanbul (a), Kaliningrad (b).

expert defined it in advance. Furthermore, we assume that the level of a node is defined as the number of nodes on the longest direct path from root to this node, plus one. As illustrated in Figure 2(b), $level(\textit{Russia}) = 2$ and $level(\textit{Kaliningrad}) = 3$.

4. LOCATION DETERMINATION STRATEGIES

The research question addressed in this paper is twofold: we want to continuously determine the location of an object while reducing the energy requirements for positioning, and we want to offer location information that is appropriate for different application scenarios. The two extrema are: GPS polling and not measuring at all. Polling is the most energy-intensive approach and not measuring is the most imprecise "solution". We developed three approaches for calculating appropriate location information with a minimal amount of energy. The first strategy reflects the fact that memory-intensive computations require much more energy than CPU-intensive ones [8] and reduces the number of required database lookups. The other two strategies aim for reducing the number of GPS request and lookup operations in order to find the correct path from a detected node to the root of LPH. This path correctly describes the different levels of detail for the current location.

4.1 Level Dependencies

In LPH the point sets' cardinalities of locations represented by higher-level nodes are smaller than those of locations represented by the linked lower-level nodes. The assignment of a location to a set of positions uses default GIS techniques. So, a db stores border polygons, and a db lookup fetches the proper location values from the db. Hence, one location lookup requires certain memory-intensive operations. Moreover, in case of continuous location determination, queries must be performed for each movement of the requesting object. However, if we know about the (semantic) dependencies among certain levels within the location poly-hierarchy, we can reduce the amount of energy-intensive db lookups by traversing LPH.

Figure 2(b) shows that if an object is located within Kaliningrad, it is in Russia and Europe and on earth, too. So, only one comparison of location sets is necessary. For the example in Figure 2(a), this is not as trivial. The requesting object is located in Istanbul and in Turkey. However, requesting the continent information requires an additional set comparison as Istanbul belongs to both Europe and Asia. So, the number of comparisons depends on the structure of the given poly-hierarchy. Our algorithm calculates the correct path for a given location while minimising the number of comparisons. We assume that the names of the leaf and inner nodes in LPH are unique and referenced within the GIS db.

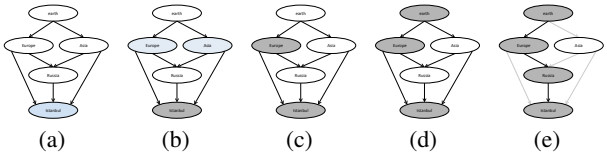


Figure 3: Traversing the location poly-hierarchy in case of Istanbul.

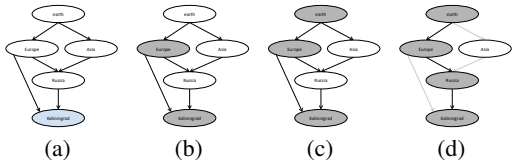


Figure 4: Traversing the location poly-hierarchy in case of Kaliningrad.

The algorithm works as follows (cf. Figures 3 and 4): At first, the proper leaf node is selected using a db query (F. 3(a); F. 4(a)). We then traverse the LPH bottom-up and mark nodes that belong to the correct path: The direct (grand)parent node(s) with the lowest level are analysed. If the current node has only one (grand)parent node in this level, this node is selected as path anchor (F. 4(b)). If more than one (grand)parent node exists on the minimal level, we check them with a db query (F. 3(b)), mark the correct one and select it as path anchor (F. 3(c)). We continue with the path anchor until we reach the root node (F. 3(d); F. 4(c)). The last step is to collect the nodes on the path from root to the leaf node including all marked inner nodes (F. 3(e); F. 4(d)).

4.2 Postponed Measurements

The postponed measurements strategy predicts the time required for a person or object to leave the current location. This time depends on the hierarchy level, the geographical model used for this level and on the movement speed [20].

For simplification purposes, the application specifies the velocity of the moving object as a movement profile (pedestrians: $\approx 6\,\mathrm{km\,h^{-1}}$, cyclists: $\approx 11\,\mathrm{km\,h^{-1}}$, car drivers: $\approx 60\,\mathrm{km\,h^{-1}}$). The most obvious approach is to take the object's current position $L_c = (x_c, y_c)$ and then calculate the minimal Euclidean distance d to all locations within the current path in LPH. For each location L on this path, we have to calculate: $\min(\sqrt{(x_c - x_l)^2 + (y_c - y_l)^2}$ $|\forall(x_l, y_l) \in L)$. With a simple calculation, we then compute the time the object would need to leave this location. If, e.g., a pedestrian is 5 km away from the city limit, he or she would need at least 50 minutes ($\frac{5000\,\mathrm{m} * 3600\,\mathrm{s}}{6000\,\mathrm{m}} = 3000\,\mathrm{s}$) to leave the city. So, we postpone the next position measurement by 50 minutes if the application requires the location at a city level. The Euclidean distance guarantees the calculation of the shortest distance. Hence, if the velocity is correct, the calculation always returns a time window the moving object must be in the current location.

This approach is used for area-like locations where no route information exists. In case of a map-based location management, we utilise crossroad data to get more precise predictions (cf. [10]). Therefore, we maintain a db with all crossroads and calculate the Euclidian distance between the current position and the closest crossroad. Of course, the prediction is more exact if we use the entire map information but storing the complete maps would require much space (e.g., for the German state of Thuringia, we have to maintain 125,034 crossroads or to store and query 657 MB of OpenStreetMap data).

Besides the postponement calculation for the various levels, we have to consider the poly-hierarchy as well. Referring back to the Istanbul-Example: if the moving object is located in Istanbul, it may take one hour to leave the city, but only 10 minutes to leave the continent. The solution for this issue is to analyse the LPH in the following way. Given the LPH, the current location and the current velocity. First we calculate the correct location path using the algorithm discussed in Section 4.1. In the next step, we calculate the postponement value for each location in this path using the appropriate calculation (Euclidian distance, crossroad analysis). The minimal value determines the time until the next measurement.

4.3 Level Appropriateness

There exist various technologies to measure positions such as geotagged Wi-Fi hotspot and cell tower databases, wireless signal triangulation/lateration, or geotagging. They vary in their energy demand and their precision. While looking at the location poly-hierarchy one can recognise that locations represented closely to the root node mostly require less precise location techniques. Country information can directly be read from the country code provided by mobile telecommunications operators. The city information can be harvested from cell tower information using a simple db lookup. We have to use the most high-energy GPS only if precise location information are required. The approach discussed in the following combines the techniques illustrated in Section 4.1 and Section 4.2.

Hierarchy level	Level	Technique
Earth	0	known to be true
Continent	1	Country code + Cell tower ID lookup
Country	2	Country code
State	3	Cell tower ID lookup
City	4	Cell tower triangulation
Street	5	GPS

Table 1: Location determination lookup table.

For the postponed measurements, we adapted the cross-level postponement value in a way that we calculate different postponement values for each level while alternating the measurement strategy (cf. Table 1). Let's say that the object is located in Istanbul, and that the GPS-based measurement resulted in a postponement value of 10 minutes for the continent, and a postponement value of 1 hour for the city. After 10 minutes, we then check the appropriateness of the continent location using energy-efficient cell tower triangulation and calculate a new postponement value for this level on this basis. Hence, in best case (we do not leave the continent), we can wait with the next GPS positioning for 50 more minutes.

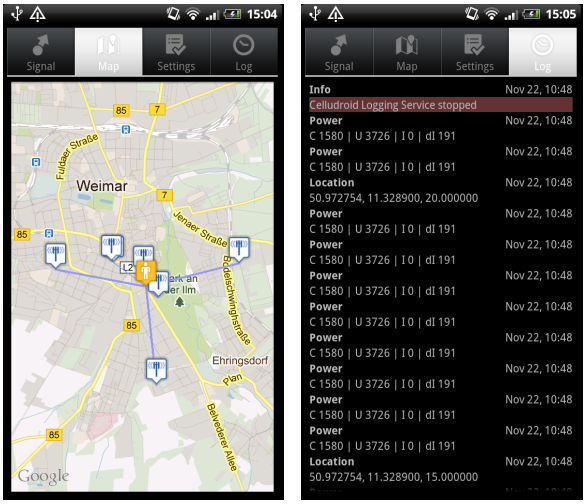
A more trivial approach supported by this idea are applications that do not need exact position information. As mentioned above, requesting the country information does not need any positioning as the information is provided by the service provider. Furthermore, less energy-intensive cell ID lookups for determining locations on the city level might be used in a polling mode. Anyway, position-based location determination such as cell tower triangulation in combination with a GIS requires too much calculation effort, if used in a polling manner. However, we will research a combination of polled and time-triggered updates of location information in a location poly-hierarchy in the near future.

5. EVALUATION

Our prototype is still work in progress. Therefore, we present a preliminary evaluation that enabled us to estimate the energy footprint of our proposed concept in an exemplary scenario.

5.1 Experimental Setup

The evaluation system was implemented as a mobile application on the Android 2.3.4 platform. We conducted our tests with the recently released HTC Sensation smartphone. As shown in Figure 5(b), the application is mainly a data logger for location and power management data. In order to gather location data, we implemented a cell tower lateration algorithm, and used the Android SDK's methods for obtaining GPS data. The application reads energy-related data (battery voltage and current) directly from the device's power management. It was implemented as an independent background logging service that gathers data even when the device is locked. The experimental setup follows our data-based energy measurement approach, as described in [12].



(a) (b)

Figure 5: Celludroid evaluation app running on an HTC Sensation smartphone.

All data was stored in an sqlite database that could later on easily be used for analysis. For convenience, the application also shows the estimated location on a map (cf. Figure 5(a)) and allows to explore the properties of nearby cell towers.

The cell tower lateration uses a Google service[1] to look the location of nearby cell towers up. Because all retrieved cell tower locations are cached in an sqlite database, subsequent location requests for previously discovered cell towers do not require additional (high-energy) network communication.

[1]Unfortunately, access to this service is not publicly documented, our implementation is based on the general process documented in this forum article: `http://stackoverflow.com/a/3356956`

Our test run consisted of a sequence of 30-minute city walks, one for each test condition:

a) baseline,

b) cell tower lateration, and

c) GPS.

In the baseline condition, the display was turned off, Wi-Fi was on, Bluetooth was off, and no background tasks except our logging service were running. During each run, power management data and cell tower locations were logged every second. GPS was requested every 5 s. However, the Android SDK cannot guarantee an exact time between GPS location updates. In fact, our measured time is slightly higher (7.2 s on average).

From the power management data, we derived electrical power and electrical energy data. In order to assess the accuracy of the gathered location data, additional processing was necessary. While the GPS data already included accuracy information, we computed accuracy information for the cell tower lateration by comparing the gathered coordinates to their counterparts from the GPS run. This enabled us to give an estimate for the upper bound of the error.

5.2 Results and Discussion

Energy

Our results indicate that cell tower triangulation has no significant impact on the device's energy demand at all. The baseline condition shows a total of 181.07 J after 10 minutes and 565.5 J after 30 minutes, while the test run with cell tower lateration resulted in a slightly higher 183.76 J after 10 minutes and 583.28 J after 30 minutes. This difference of 1.5 % is within the measuring tolerance of our method. Far more interesting is the difference between baseline

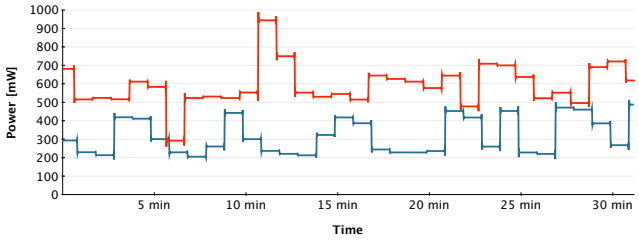


Figure 6: Comparison of power consumption for GPS (red line, top) and cell tower lateration (blue line, bottom) during continuous position determination.

and GPS condition. Figure 6 documents the power consumption progress during cell tower and GPS run. In the GPS run, the device required 316.07 J after 10 minutes and 1057.8 J after 30 minutes. Compared to baseline, this is an increase of 74.56 % after 10 minutes, or 87.06 % after 30 minutes (cf. Figure 7). Remember, GPS data was gathered every 7.2 s on average. In these 7.2 s, our setup required 4896 mJ of energy on average ($7.2\,\text{s} \cdot 680\,\text{mJ}\,\text{s}^{-1}$). In the baseline condition, 7.2 s of idling required 1296 mJ of energy on average ($7.2\,\text{s} \cdot 180\,\text{mJ}\,\text{s}^{-1}$), which is 26.5 % of the amount required for GPS.

Accuracy

The results regarding accuracy of the position determination techniques are very clear. While the GPS condition results show an average of 9.2 m, cell tower lateration performs drastically worse at 308.26 m (a difference of 3242.55 %). Some outliers in the data

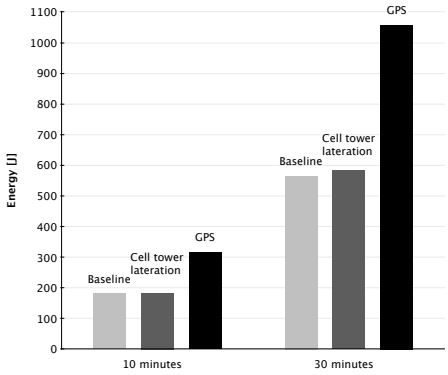


Figure 7: Comparison of accumulated energy demand.

even showed an error of more than 800 m. These extreme differences highlight the fact that the reduced energy requirements of cell tower lateration condition have an at least equally drastic influence on accuracy.

5.3 Scenario

The gathered results provide insight into the areas of application where sufficient potential exists to reduce the energy requirements for continuous location determination. In this subsection, we sketch a scenario that relies on our three conceptual pillars:

a) using level dependencies,

b) postponed measurements,

c) using appropriate positioning techniques.

This scenario-based evaluation surely cannot serve as a proof for our proposed concept, but it provides a glimpse on its possible impact. Our scenario application is a mobile tourist information system for smartphones. All data is stored on the mobile device and can be accessed using db queries. The application can access the following device's location sensors:

a) SIM card operator's country code (country information),

b) Cell tower association (state information),

c) Cell tower lateration (city part/street information),

d) GPS (street number information).

In our scenario, a tourist from Japan is on a bus tour through Germany and is currently visiting Thuringia. In Weimar, she decides to start using the tourist information system.

Using level dependencies

Upon first use, the system requires a single positioning with cell tower lateration. With the gathered coordinates, the country, state, and city nodes of the location poly-hierarchy can be determined: earth→Europe→Germany→Thuringia→Weimar. The system uses this data to switch to the tour mode for Thuringia.

Using appropriate techniques according to level

In this mode, a continuous perimeter search delivers points of interest, such as restaurant, museums, public places, or theaters. Because this feature at this point only requires the general information about the presence of such points of interest (and not a detailed routing information on how to get there), it is appropriate to rely on cell tower lateration.

Postponing unnecessary measurements

When the tourist finished exploring the city, she wants to find a nice place to have lunch. After selecting one of the restaurants from the perimeter search result list, the tourist information system enters the navigation mode. In this mode, a database of intersections is used in combination with a movement profile to estimate important turning points along a route where it is necessary to activate the GPS technique.

Estimation of reduced energy requirements

Upon first use, at least one GPS request can be saved. As we have shown in the previous subsection, each request requires about 4896 mJ. The continuous perimeter search for points of interest took 2 hours in our scenario. During these 2 hours, a continuous GPS polling would have required about 1000 GPS requests (4896 J). In contrast, the cell tower lateration used by the system only required 1296 J. Giving an estimation of the reduction achieved by the postponed measurements in the navigation mode (using the intersection database) is very difficult, as it relies on a multitude of factors (e.g., velocity, distance between intersections). A conservative lower bound would be to assume that this method reduces the amount of GPS requests to 50 %.

6. SUMMARY, CONCLUSIONS, AND OUTLOOK

We addressed two research questions in the area of continuous location determination: We analysed precision appropriateness and discussed energy requirements. We utilised the poly-hierarchical nature of locations to provide different levels of location information. We discussed three approaches: We reduced the amount of location calculations within the location poly-hierarchy, we introduced the concept of postponed measurements, and we discussed appropriateness issues in location poly-hierarchy levels.

The overall goal behind our research is to realise a location framework that application developers can use to implement location-aware applications without the need to take care of the high energy requirements of GPS. We know that various frameworks do exist and address the same issue. However, our results show the potential to reduce the required energy further. The paper opens various research directions that we will follow on: The concept of location poly-hierarchies requires a detailed and formal definition. While location hierarchies and ontology-based models are well researched, poly-hierarchies are rather novel. Furthermore, more research needs to be done in the area of calculating the postponement values, and the combination of polling and postponed updates must be addressed.

7. REFERENCES

[1] C. Becker and F. Dürr. On location models for ubiquitous computing. *Personal and Ubiquitous Computing*, 9(1):20–31, 2005.

[2] M. Beigl, T. Zimmer, and C. Decker. A location model for communicating and processing of context. *Personal Ubiquitous Computing*, 6:341–357, Jan. 2002.

[3] C. Bunse and H. Höpfner. Resource substitution with components — optimizing energy consumption. In *ICSOFT '08 Proc.*, volume SE/GSDCA/MUSE, pages 28–35. INSTICC press, July 2008.

[4] I. Constandache, S. Gaonkar, M. Sayler, R. R. Choudhury, and L. P. Cox. Energy-Efficient Localization via Personal Mobility Profiling. In *MobiCASE '09 Proc.*, pages 203–222. Springer, 209.

[5] Z. Dongqing, L. Zhiping, and Z. Xiguang. Location and its Semantics in Location-Based Services. *Geo-spatial Information Science*, 10(2):145–150, June 2007.

[6] F. Dürr and K. Rothermel. On a location model for fine-grained geocast. In *UbiComp '03 Proc.*, pages 18–35. Springer, 2003.

[7] I. Heywood, S. Cornelius, and S. Carver. *An Introduction to Geographical Information Systems*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, Nov. 2002.

[8] H. Höpfner and C. Bunse. Towards an energy-consumption based complexity classification for resource substitution strategies. In *GVDB '10 Proc.*, volume 581. CEUR-WS.org, May 2010.

[9] H. Höpfner and C. Bunse. Energy Awareness Needs a Rethinking in Software Development. In *ICSOFT '11 Proc.*, volume 2, pages 294–297. SciTePress, 2011.

[10] H. Höpfner and M. Schirmer. Energy efficient continuous location determination for pedestrian information systems. In *MobiDE '12 Proceedings*, 2012. accepted for publication.

[11] H. Höpfner and M. Schirmer. Towards modeling locations as poly-hierarchies. In *GVBD '12 Proc.*, May 2012. accepted for publication, forthcoming.

[12] H. Höpfner, M. Schirmer, and C. Bunse. On measuring smartphones' software energy requirements. In *ICSOFT '12 Proc.*, 2012. accepted for publication, forthcoming.

[13] S. Jevtic, M. Kotowsky, R. P. Dick, P. A. Dinda, and C. Dowding. Lucid dreaming: Reliable analog event detection for energy-constrained applications. In *IPSN 'ß07 Proc.*, pages 350–359. ACM, 2007.

[14] C. Jiang and P. Steenkiste. A hybrid location model with a computable location identifier for ubiquitous computing. In *Ubicomp '02 Proc.*, pages 246–263. Springer, 2002.

[15] T. Kauppinen, R. Henriksson, R. Sinkkilä, R. Lindroos, J. Väätäinen, and E. Hyvönen. Ontology-based Disambiguation of Spatiotemporal Locations. In *IRSW '08 Proc.*, volume 422. CEUR-WS.org, 2008.

[16] B. N. Schilit, A. LaMarca, G. Borriello, W. G. Griswold, D. McDonald, E. Lazowska, A. Balachandran, J. Hong, and V. Iverson. Challenge: ubiquitous location-aware computing and the "place lab" initiative. In *WMASH '03 Proc.*, pages 29–35. ACM, 2003.

[17] M. Schirmer and H. Höpfner. SenST: Approaches for Reducing the Energy Consumption of Smartphone-Based Context Recognition. In *CONTEXT '11 Proc.*, pages 250–263. Springer, 2011.

[18] A. Y. Seydim, M. H. Dunham, and V. Kumar. Location dependent query processing. In *MobiDE '01 Proc.*, pages 47–53. ACM, 2001.

[19] J. P. Sousa, R. K. Balan, V. Poladian, D. Garlan, and M. Satyanarayanan. User Guidance of Resource-Adaptive Systems. In *ICSOFT '08 Proc.*, volume SE/GSDCA/MUSE, pages 36–44. INSTICC press, 2008.

[20] O. Wolfson, B. Xu, S. Chamberlain, and L. Jiang. Moving Objects Databases: Issues and Solutions. In *SSDBM '98 Proc.*, pages 111–122. IEEE CS, 1998.

[21] J. Ye, L. Coyle, S. Dobson, and P. Nixon. A unified semantics space model. In *LoCA '07 Proc.*, LNCS, pages 103–120. Springer, 2007.

[22] P. A. Zandbergen. Accuracy of iPhone Locations: A Comparison of Assisted GPS, WiFi and Cellular Positioning. *Transactions in GIS*, 13(s1):5–26, July 2009.