Infocommunications Journal

A PUBLICATION OF THE SCIENTIFIC ASSOCIATION FOR INFOCOMMUNICATIONS (HTE)

December 2014 Volume VI Number 4 ISSN 2061-2079

SPECIAL ISSUE ON FUTURE INTERNET, PART II

PAPER FROM OPEN CALL

CALL FOR PAPERS

ADDITIONAL

Technically Co-Sponsored by

IEEE
COMMUNICATIONS
SOCIETY

hte

IEEE
HUNGARY SECTION

Editorial Board

Indexing information

Infocommunications Journal is covered by Inspec, Compendex and Scopus.

Infocommunications Journal

Technically co-sponsored by IEEE Communications Society and IEEE Hungary Section

Supporters

GÁBOR BÓDI – president, National Council for Telecommunications and Informatics (NHIT)
GÁBOR MAGYAR – president, Scientific Association for Infocommunications (HTE)

Editorial Office (Subscription and Advertisements)**:**
Scientific Association for Infocommunications
H-1051 Budapest, Bajcsy-Zsilinszky str. 12, Room: 502
Phone: +36 1 353 1027, Fax: +36 1 353 0451
E-mail: info@hte.hu • Web: www.hte.hu

Articles can be sent also to the following address:
Budapest University of Technology and Economics
Department of Networked Systems and Services
Tel.: +36 1 463 3261, Fax: +36 1 463 3263
E-mail: szabo@hit.bme.hu

Subscription rates for foreign subscribers: 4 issues 50 USD, single copies 15 USD + postage

Publisher: PÉTER NAGY • Manager: ANDRÁS DANKÓ

HU ISSN 2061-2079 • Layout: MATT DTP Bt. • Printed by: FOM Media

Special Issue on the Future Internet, Part II

Gyula Sallai, Wolfgang Schreiner, and János Sztrik

Recent dramatic changes such as the rising number of Internet users, the penetration of portable and mobile devices, or the Internet of Things, has motivated a number of research initiatives, labeled "Future Internet" worldwide, supported by NSF in the USA and EU research framework programs in Europe. In Hungary, the "Future Internet Research, Services and Technology - FIRST" project, supported by the European Social Funds, focuses on key theoretical, modeling, planning, application and experimental aspects of Future Internet. The nine papers published in two subsequent special issues of this journal, demonstrate the research results achieved by the FIRST research community in various fields related to Future Internet. Six papers were published in Issue 3, 2014, and further three papers are published in this issue.

The authors of the first paper, "Correlation clustering of graphs and integers", *Shigeki Akiyama, László Aszalós, Lajos Hajdu, Attila Pethő,* considered the problem of correlation clustering from three different but closely related aspects. First, new results are presented that have been derived for the graph model of the problem, considering an increasing family of graphs. Then particular sets with a specific relation have been investigated. Finally, the set of so-called S-units have been considered, under the same relation as for positive integers. Here the authors have proved that in contrast with the case of positive integers, after some point the optimal clustering is always given by the trivial clustering (consisting of a single class).

Authors *Károly Farkas, Gábor Fehér, András Benczúr, and Csaba Sidló,* in their paper "Crowdsensing Based Public Transport Information Service in Smart Cities", introduce the XMPP based communication framework that was designed to facilitate the development of crowd assisted smart city applications. Then the crowdsensing based real-time public transport information service is presented, implemented on top of the framework, and its front-end Android application, called TrafficInfo, in detail, together with the stop event detector developed by the authors. This detector automatically detects halt events of public transport vehicles at the stops.

The survey paper by *Péter Battyányi and György Vaszil,* titled "Membrane Systems from the Viewpoint of the Chemical Computing Paradigm", deal with membrane systems that are nature motivated abstract computational models inspired by basic features of biological cells and their membranes. The authors first review some of the basic features and properties of the chemical paradigm of computation, and also give a short introduction to membrane systems. Then they examine the relationship of the certain chemical programming formalisms and some simple types of membrane systems.

Guest Editors:

GYULA SALLAI received MSc degree from the Budapest University of Technology and Economics (BME), PhD and DSc degrees from the Hungarian Academy of Sciences (MTA), all in telecommunications. He was senior researcher in telecommunication network planning, then research director, strategic director, later deputy CEO with the Hungarian Telecom Company; then international vice president, after that executive vice president for the ICT regulation with the Communication Authority of Hungary. From 2002 to 2010 he was the head of the Department of Telecommunications and Media Informatics of the BME, and from 2004 to 2008 the vice-rector of the BME as well. From 2005 to 2011 he was also the chairman of the Telecommunication Committee of the MTA and the president of the Hungarian Scientific Association for Infocommunications (HTE).
Recently he is full-professor at the BME, Scientific Director of Future Internet Research Coordination Centre, member of the FIRST Project Council and honorary president of the HTE. His main fields of interest are the ICT trends, strategic, management and regulatory issues, Future Internet engineering.

WOLFGANG SCHREINER is since 2004 associate professor of the Research Institute for Symbolic Computation (RISC) at the Johannes Kepler University Linz, Austria. His research areas are formal methods and parallel and distributed computing; he has directed in these areas various national and international research and development projects, participated in the program committees of 90 conferences, served as evaluator for various European projects, and is member of the editorial board of the Journal of Universal Computer Science.
Prof. Schreiner has (co-)authored 13 book chapters, 9 journal publications, 46 other refereed publications, 12 non-refereed publications and 70 technical reports.

JÁNOS SZTRIK is a Full Professor at the Faculty of Informatics and Head of Department of Informatics Systems and Networks, University of Debrecen, Debrecen, Hungary. He received the M.Sc. degree in 1978, the Ph.D in 1980 both in probability theory and mathematical statistics from the University of Debrecen. He obtained the Candidate of Mathematical Sciences degree in probability theory and mathematical statistics in 1989 from the Kiev State University, Kiev, USSR, habilitation from University of Debrecen in 1999, Doctor of the Hungarian Academy of Sciences, Budapest, 2002.
His research interests are in the field of production systems modeling and analysis, queueing theory, reliability theory, and computer science.

May 20-22, 2015

Call for papers

Budapest, Hungary

EUROPEAN WIRELESS 2015

ew2015.european-wireless.org

The European Wireless (EW) conference is a key venue for European researchers to get in touch with the latest trends in wireless communications and networking. The 21th EW conference will take place in Budapest, the lively capital of Hungary, organized by the Budapest University of Technology and Economics (BME). The main theme of EW 2015 will be "5G and beyond."

Topics of interest include, but are not limited to, the following:

FW Fundamental Wireless

- ○ Modulation and coding for wireless communications
- ○ Signal processing for wireless communications
- ○ Wireless models, synchronization, estimation, equalization
- ○ MIMO systems, space-time coding, diversity
- ○ Fundamental limits, information theory for wireless
- ○ Multiple access schemes, multiuser detection
- ○ Interference mitigation and management
- ○ Distributed coding and cooperative diversity
- ○ Localization and positioning in wireless systems
- ○ Source and joint source/channel coding
- ○ Spectrum sensing and wireless parameter estimation

TW Technology for Wireless

- ○ Migration, integration, and convergence towards 5G
- ○ WiFi, LTE, 3GPP, Heterogeneous Networks
- ○ Wireless LAN/PAN/BAN, Ad Hoc, Mesh networks
- ○ Near-field communications & RFID
- ○ Wired-wireless integration
- ○ Ultra-Wideband (UWB) communications
- ○ Mm-wave communications
- ○ Wireless sensors & actuators networks
- ○ Vehicular and disruption tolerant wireless networks
- ○ Optical wireless and visible light communications

EW Efficient Wireless

- ○ Power: green communications, energy harvesting devices
- ○ Spectrum: cognitive radio, spectrum-aware techniques
- ○ Implementation: low-complexity and scalable systems
- ○ Reliability: robust and dependable wireless systems
- ○ Cost: low-cost radio, sustainable wireless
- ○ Security: privacy and trust in wireless networks

AW Advanced Wireless

- ○ Protocols and architectures for wireless networks
- ○ Channel coding, error protection, network coding
- ○ Cross-layer issues in wireless networks
- ○ Cognitive radio for wireless communications
- ○ QoS and resource allocation in wireless networks
- ○ Mobile/wireless networks modeling and simulation
- ○ Localization and positioning in wireless scenarios
- ○ Optimization and game theory for wireless
- ○ Topology control, self-organizing wireless networks
- ○ Transport layer for wireless communications
- ○ Relays and buffers in wireless networks
- ○ Tools for modeling and analysis of wireless systems

PW Practical Wireless

- ○ Implementation issues in wireless systems
- ○ Testbeds and experimental systems
- ○ Antenna and RF modeling and design
- ○ Mobility management and billing technologies
- ○ Mobile apps and platforms
- ○ Regulation and standardization for wireless
- ○ Context awareness
- ○ Emerging applications in wireless networks

VW Vision for Wireless

- ○ Personal wireless communications beyond 5G
- ○ Software defined wireless networks and re-configurability
- ○ M2M communications and the Internet of Things
- ○ Storage, smart caching, and cloud for wireless
- ○ Wireless social networks, participatory computing
- ○ Molecular and nano-scale wireless communications
- ○ New disruptive concepts for wireless systems

General chair: Hassan Charaf (BME, HU)
General co-chair: Marcos Katz (University of Oulu, FI)
TPC chair: Leonardo Badia (University of Padova, IT) and Mischa Dohler (King's College London, UK)
Steering committee chair: Frank Fitzek (Aalborg University, DK)
Tutorial chairs: Sergio Palazzo (University of Catania, IT) and Morten V. Pedersen (Aalborg University, DK)
Workshops: Christian Weitfield (TU Dortmund, DE), Péter Ekler (BME) and László Lengyel (BME).
Publicity chair: Stefan Valentin (Alcatel Lucent, DE) and Leonardo Militano (Mediterranea University of Reggio Calabria, IT)
Financial chair: Volker Schanz (VDE ITG, DE)
Secretariat/Registration: Christina Gaußmann (VDE ITG, DE)
BME local committee: István Vajk, János Levendovszky, Sándor Imre, Bertalan Forstner, László Lengyel, Péter Ekler, Imre Kelényi, József Bíró, János Tapolcai, Attila Vidács and Rolland Vida

The EW conference is committed to high publication ethics standards through a rigorous single-blind peer-review process. Submitted manuscripts must be original and not published or under consideration elsewhere. They must not infringe any copyright or third party right. Proceedings of EW 2015 will be available on IEEEXplore and Scopus (approval pending). Authors of selected papers will be invited to submit a journal extended version for a special issue of Wiley Transactions on Emerging Telecommunications Technologies.

Important dates

paper submission: February 2, 2015
notification of acceptance: March 22, 2015
camera ready due: March 29, 2015

Correlation clustering of graphs and integers

S. Akiyama, L. Aszalós, L. Hajdu, A. Pethő

***Abstract*—Correlation clustering can be modeled in the following way. Let A be a nonempty set, and $\sim$ be a symmetric binary relation on A. Consider a partition (clustering) $\mathcal{P}$ of A. We say that two distinct elements $a, b \in A$ are in conflict, if $a \sim b$, but a and b belong to different classes (clusters) of $\mathcal{P}$, or if $a \not\sim b$, however, these elements belong to the same class of $\mathcal{P}$. The main objective in correlation clustering is to find an optimal $\mathcal{P}$ with respect to $\sim$, i.e. a clustering yielding the minimal number of conflicts. We note that correlation clustering, among others, plays an important role in machine learning.**

In this paper we provide results in three different, but closely connected directions. First we prove general new results for correlation clustering, using an alternative graph model of the problem. Then we deal with the correlation clustering of positive integers, with respect to a relation $\sim$ based on coprimality. Note that this part is in fact a survey of our earlier results. Finally, we consider the set of so-called S-units, which are positive integers having all prime divisors in a fixed finite set. Here we prove new results, again with respect to a relation defined by the help of coprimality. We note that interestingly, the shape of the optimal clustering radically differs for integers and S-units.

***Index Terms*—correlation clustering, graphs, integers, S-units.**

I. Introduction

Correlation clustering was introduced in the field of machine learning. We refer to the paper of Bansal et al. [3], which also gives an excellent overview of the mathematical background. Let G be a complete graph on n vertices and label its edges with $+1$ or -1 depending on whether the endpoints have been deemed to be similar or different. Consider a partition of the vertices. Two edges are in conflict with respect to the partition if they belong to the same class, but are different, or they belong to different classes although they are similar. The ultimate goal of correlation clustering is to find a partition with minimal number of conflicts. The special feature of this clustering is that the number of clusters is not specified. In some applications G is not necessarily a complete graph like in [5] or the labels of the edges are real numbers like in [9].

Correlation clustering admits the following equivalent model too. Let A be a nonempty set, $\sim$ be a tolerance relation on A, i.e., a reflexive and symmetric binary relation. Consider a partition (clustering) $\mathcal{P}$ of A. We say that two elements $a, b \in A$ are in conflict, if $a \sim b$, but a and b belong to different classes (clusters) of $\mathcal{P}$, or if $a \not\sim b$, however, these elements belong to the same class of $\mathcal{P}$. The main objective is to find an optimal $\mathcal{P}$ with respect to $\sim$, i.e. a clustering yielding the minimal number of conflicts. It is worth to mention that if we also assume that $\sim$ is transitive, then it is an equivalence relation. In this case the optimal clustering is obviously provided by the equivalence classes of $\sim$. So this is the lack of the transitive property which makes the problem of correlation clustering interesting and important. Every clustering of A implies an equivalence relation on A. The number of conflicts in a clustering reflects a kind of distance of $\sim$ to this equivalence relation. An optimal correlation clustering causes the least number of conflicts among all clusterings, thus it induces a nearest equivalence relation to $\sim$.

A typical application of correlation clustering is the classification of unknown topics of (scientific) papers. In this case the papers represent the elements of A and two papers are considered to be similar (or being in relation $\sim$), if one of them refers to the other. The classes of an optimal clustering then can be interpreted as the topics of the papers. This kind of clustering has many applications: image segmentation [15], identifying biologically relevant groups of genes [4], examining social coalitions [16], reducing energy consumption in wireless sensor networks [6], modeling physical processes [12], etc.

The number of partitions of sets having n elements grows exponentially, so the exhaustive search is not available to find an optimal clustering. Bansal et al. [3] showed that to find an optimal clustering is NP-hard. Beside this, they also proposed and analyzed algorithms for approximate solutions of the problem. In fact the correlation clustering can be considered to be an optimization problem: one should find the clustering minimizing the number of conflicts. Thus it is possible to apply traditional and modern optimization algorithms to find almost optimal clusterings. Following this approach, Bakó and Aszalós [2] have implemented several traditional methods, and have also invented some new ones.

In this paper we consider infinite growing sequences of labeled graphs such that the labeling is hereditary (see Section II). Then we can define lower and upper densities of edges with label $+1$ as well as of the classes in an optimal correlation clustering. The aim of Section II is to show relations between these quantities. Our results show that the choice of the labeling heavily affects the structure of the optimal clustering. For example Theorem 1 implies that if the upper density of edges with $+1$ is less than $1/2$ then there are at least two classes in an optimal correlation clustering. The value $1/2$ is the best possible by Remark 1.

In Sections III and IV we investigate particular examples. To introduce them we switch to the relational model. In that case we may assume that $A_i, i = 1, 2, \dots$ is a chain of subsets of $\mathbb{N}$ and $\sim_i$ is the restriction of $\sim$ to A_i. Here $\sim$ denotes a reflexive and symmetric relation on $\mathbb{N}$. After fixing the basic

Manuscript received September 11, 2014, revised December 8, 2014.

S. Akiyama is with the Institute of Mathematics, University of Tsukuba, 1-1-1 Tennodai, Tsukuba, Ibaraki, Japan (zip:350-8571)

L. Aszalós is with the Faculty of Computer Science, University of Debrecen, H-4010 Debrecen, P.O. Box 12, Hungary

L. Hajdu is with the Institute of Mathematics, University of Debrecen, H-4010 Debrecen, P.O. Box 12, Hungary

A. Pethő is with the Faculty of Computer Science, University of Debrecen, H-4010 Debrecen, P.O. Box 12, Hungary

set to $\mathbb{N}$ it is natural to use the coprimality to define the relation $\sim$. More precisely, for positive integers $a, b \in A$ we set $a \sim b$ if $\gcd(a, b) > 1$ or $a = b = 1$. In Section III we consider this relation with sets A_n of positive integers not exceeding n ($n = 1, 2, \dots$). The results of this section are published in the paper [1], so we only outline the main results and methods here. We present a natural greedy algorithm, Algorithm 1, which computes locally optimal clustering and prove that it behaves regularly for $n < n_0 = 3{\cdot}5{\cdot}7{\cdot}11{\cdot}13{\cdot}17{\cdot}19{\cdot}23 = 111\,546\,435$, but from n_0 on this regularity disappears. Further we show that its optimal correlation clustering has at least two classes. In Section IV we give a similar analysis, but for the sets of S-units (or generalized Hamming numbers) not exceeding n ($n = 1, 2, \dots$). The results presented here are all new. We show that the optimal correlation clustering is in this case asymptotically trivial, i.e. has only one class. Although the asymptotic result is smooth, there are usually many growing classes in the early stages, but after a while the largest class starts to collect all elements like a black hole.

Finally, we give concluding remarks in Section V.

II. Correlation clustering of graphs

In this section we consider the problem of correlation clustering not for a single graph, but for an increasing family of graphs. For $n \geq 1$, let K_n be the complete graph of n vertices. Write $V(K_n)$ and $E(K_n)$ for the set of vertices and edges of K_n, respectively. Take an arbitrary labeling

$$c_n : \ E(K_n) \to \{-1, 1\}$$

of the edges of K_n, subject to the hereditary (consistency) condition that for some embedding

$$\sigma_{n-1} : K_{n-1} \to K_n$$

of K_n into K_{n-1} the coloring is invariant, that is

$$c_{n-1}(e) = c_n(\sigma_{n-1}(e)) \text{ for } e \in E(K_{n-1}).$$

Thus

$$K_1 \xrightarrow{\sigma_1} K_2 \xrightarrow{\sigma_2} \dots$$

can be considered as an increasing sequence of labeled graphs. We define the upper and lower densities of the edges having label 1 in the usual way:

$$\overline{g} = \limsup_{n\to\infty} \frac{|\{e \in E(K_n) \ : \ c_n(e) = 1\}|}{|E(K_n)|} =$$
$$= \limsup_{n\to\infty} \frac{|\{e \in E(K_n) \ : \ c_n(e) = 1\}|}{n(n-1)/2}$$

and

$$\underline{g} = \liminf_{n\to\infty} \frac{|\{e \in E(K_n) \ : \ c_n(e) = 1\}|}{|E(K_n)|} =$$
$$= \liminf_{n\to\infty} \frac{|\{e \in E(K_n) \ : \ c_n(e) = 1\}|}{n(n-1)/2}.$$

Here and later on, $|H|$ denotes the number of elements of the set H. Let $\mathcal{P}(n)$ be an optimal clustering of (K_n, c_n), with classes $\mathcal{P}_1(n), \mathcal{P}_2(n), \dots, \mathcal{P}_{m(n)}(n)$. Here without loss of generality we may assume that the classes are arranged in non-increasing order with respect to cardinality, that is

$$|\mathcal{P}_j(n)| \geq |\mathcal{P}_{j+1}(n)| \ (j = 1, \dots, m(n) - 1).$$

Define the upper and lower cluster densities of $\mathcal{P}_j(n)$ by

$$\overline{\rho}_j = \limsup_{n\to\infty} \frac{|\mathcal{P}_j(n)|}{n}, \qquad \underline{\rho}_j = \liminf_{n\to\infty} \frac{|\mathcal{P}_j(n)|}{n}$$

for $j = 1, \dots, m(n)$. If $j > m(n)$ then we set $\mathcal{P}_j(n) = \emptyset$. Clearly we have $\overline{\rho}_j \geq \overline{\rho}_{j+1}$ and $\underline{\rho}_j \geq \underline{\rho}_{j+1}$ for $j \geq 1$.

Theorem 1. *We have*

$$\sum_i \underline{\rho}_i^2 \leq 2\underline{g}, \qquad \overline{\rho_1}^2 \leq 2\overline{g}$$

and

$$\overline{g} - \sum_{i<j} \overline{\rho}_i \overline{\rho}_j \leq \sum_i \overline{\rho}_i^2, \qquad \underline{g} - \sum_{i<j} \underline{\rho}_i \underline{\rho}_j \leq \sum_j \underline{\rho}_i^2.$$

Proof. We claim that $\sum_i \underline{\rho}_i \leq 1$. In fact, for any $m \in \mathbb{N}$ and any $\varepsilon > 0$, there exists an $n_0 \in \mathbb{N}$ such that

$$\underline{\rho}_j - \varepsilon/m \leq |\mathcal{P}_j(n)|/n$$

for $j \leq m$ and $n \geq n_0$. Thus

$$\sum_{j=1}^m \underline{\rho}_j \leq \sum_{n=1}^m \frac{|\mathcal{P}_j(n)|}{n} + \varepsilon \leq 1 + \varepsilon.$$

As one can choose ε and m arbitrarily, the above inequality proves our claim. This fact is used in the last part of the proof of the first inequality.

As $\mathcal{P}_j(n)$ is an optimal cluster, in the induced graph to $\mathcal{P}_j(n)$ of (K_n, c_n), at least half of the edges belonging to each vertex of $\mathcal{P}_j(n)$ must have label 1. Indeed, if this does not hold for some vertex v of $\mathcal{P}_j(n)$, then the cluster $\mathcal{P}_j(n)$ can be divided into $\mathcal{P}_j(n) \setminus \{v\}$ and $\{v\}$, and in the new clustering the number of conflicts is less. This implies that among $|E(\mathcal{P}_j(n))|$ edges, there are at least $|E(\mathcal{P}_j(n))|/2$ edges with label 1. From the inequality

$$\frac{1}{2} \sum_j |E(\mathcal{P}_j(n))| \leq |\{e \in E(K_n) \ : \ c_n(e) = 1\}|, \qquad (1)$$

for any $m \in \mathbb{N}$ and $\varepsilon_1 > 0$, there exists an $n_1 \in \mathbb{N}$ such that

$$\frac{1}{2} \sum_{j=1}^m \frac{((\underline{\rho}_j - \varepsilon_1)n)((\underline{\rho}_j - \varepsilon_1)n - 1)}{2} \leq$$
$$\leq |\{e \in E(K_n) \ : \ c_n(e) = 1\}|$$

for $n \geq n_1$. Thus for any $\varepsilon_2 > 0$

$$\frac{1}{2} \sum_{j=1}^m \frac{((\underline{\rho}_j - \varepsilon_1)n)((\underline{\rho}_j - \varepsilon_1)n - 1)}{2} \leq (\underline{g} + \varepsilon_2)\frac{n(n-1)}{2}$$

holds for infinitely many n. Dividing by $n^2/2$ and letting n tend to ∞, we obtain the first inequality, since m, ε_1, and ε_2 are arbitrary.

It is also clear from (1) that for any $\varepsilon_1 > 0$ and $\varepsilon_2 > 0$, we have

$$\frac{((\overline{\rho}_1 - \varepsilon_1)n)((\overline{\rho}_1 - \varepsilon_1)n - 1)}{2} \leq$$

$$\leq |\{e \in E(K_n) \ : \ c_n(e) = 1\}| \leq (\overline{g} + \varepsilon_2)\frac{n(n-1)}{2}$$

for infinitely many n, giving the second inequality.

Consider now two clusters $\mathcal{P}_i(n)$ and $\mathcal{P}_j(n)$. Among the $|\mathcal{P}_i(n)| \cdot |\mathcal{P}_j(n)|$ edges joining the two clusters in K_n, the number of edges labeled by 1 can be at most $|\mathcal{P}_i(n)| \cdot |\mathcal{P}_j(n)|/2$. Indeed, otherwise we could decrease the number of conflicts by taking $\mathcal{P}_i(n) \cup \mathcal{P}_j(n)$ as a new cluster. So

$$\frac{1}{2}\sum_{i<j} |\mathcal{P}_i(n)| \cdot |\mathcal{P}_j(n)|$$

is an upper bound for the number of edges labeled by 1, connecting two distinct clusters $\mathcal{P}_i(n)$ and $\mathcal{P}_j(n)$. On the other hand, for any $\varepsilon_3 > 0$ and for infinitely many n, there exist at least

$$(\overline{g} - \varepsilon_3)|E(K_n)| - \sum |E(\mathcal{P}_j(n))|$$

edges labeled by 1, connecting two distinct clusters $\mathcal{P}_i(n)$ and $\mathcal{P}_j(n)$. Thus we get the inequality

$$\frac{1}{2}\sum_{i<j} |\mathcal{P}_i(n)| \cdot |\mathcal{P}_j(n)| \geq$$

$$\geq (\overline{g} - \varepsilon_3)|E(K_n)| - \sum |E(\mathcal{P}_j(n))|$$

for infinitely many n. Therefore for any $\varepsilon_2 > 0$, we have

$$\frac{1}{2}\sum_{i<j} ((\overline{\rho}_i + \varepsilon_2)n)((\overline{\rho}_j + \varepsilon_2)n) \geq$$

$$\geq (\overline{g} - \varepsilon_3)\frac{n(n-1)}{2} - \sum_j \frac{((\overline{\rho}_j + \varepsilon_2)n)(((\overline{\rho}_j + \varepsilon_2)n) - 1)}{2}$$

for infinitely many n. This implies the third inequality. The proof of the last inequality is similar, and our statement follows. □

Corollary 1. *Using the previous notation, we have*

$$\sqrt{\overline{g}} \leq \sum_j \overline{\rho}_j \quad \textit{and} \quad \sqrt{\underline{g}} \leq \sum_j \underline{\rho}_j.$$

Proof. The first assertion follows from

$$\overline{g} \leq \overline{g} + \sum_{i<j} \overline{\rho}_i \overline{\rho}_j \leq \left(\sum_j \overline{\rho}_j\right)^2$$

using the third inequality of Theorem 1. The proof of the second inequality is similar. □

We say that the clusters are *full* if $\sum_{j=1}^{\infty} \underline{\rho}_j = 1$.

Not all clusterings are full. For example, we may introduce an ordering of vertices of K_n and let $\mathcal{P}_1(n)$ be the first half of the vertices and remaining $\mathcal{P}_j(n)$ be singletons for $j \geq 2$. Then we have $\underline{\rho}_1 = 1/2$ and $\underline{\rho}_j = 0$ for $j \geq 2$.

Corollary 2. *Assume that the optimal clusters are full. If $g = \overline{g} = \underline{g}$ then we have*

$$1/2 - \sum_{i<j} \underline{\rho}_i \underline{\rho}_j \leq g \leq 1 - \sum_{i<j} \underline{\rho}_i \underline{\rho}_j.$$

In particular, $\underline{\rho}_1 = 1$ and $\underline{\rho}_j = 0$ for $j > 1$ holds if and only if $g = 1$.

Proof. The statement follows from

$$g + \sum_{i<j} \underline{\rho}_i \underline{\rho}_j \leq \left(\sum_j \underline{\rho}_j\right)^2 \leq 2g + 2\sum_{i<j} \underline{\rho}_i \underline{\rho}_j$$

and $\sum_j \underline{\rho}_j = 1$. Since $\underline{\rho}_{j+1} \geq \underline{\rho}_j$ by definition, the inequality shows that $\underline{\rho}_1 \underline{\rho}_2 > 0$ holds if and only if $g < 1$. □

The last statement shows that the edges labeled by 1 must be of density 1 in order to have only one non-empty class (namely, the whole K_n) in an optimal clustering. At this point we need to introduce some new notions.

A graph G is *locally stable* if the degree of each vertex is at least $\lceil(|G| - 1)/2\rceil$. It is *globally stable* if for any partition $V(G) = A \cup B$ (disjoint), there are at least $\lceil |A| \cdot |B|/2 \rceil$ edges connecting A and B. Denote by $G(\mathcal{P}_j(n))$ the graph obtained from the graph induced by the optimal cluster $\mathcal{P}_j(n)$ of (K_n, c_n), by removing all its edges labeled by -1. Then $G(\mathcal{P}_j(n))$ is globally stable, since otherwise a corresponding partition $A \cup B$ gives a lower number of conflicts. For brevity, we say that $\mathcal{P}_j(n)$ is globally stable if $G(\mathcal{P}_j(n))$ has this property.

Theorem 2. *If K_n is a single cluster, then its global stability implies that this is an optimal clustering (consisting of one cluster).*

Proof. Assume that $K_n = \mathcal{P}$ is globally stable and consider a different partition

$$K_n = \bigcup_{i=1}^{\ell} Q_i.$$

Let $c(Q_i, Q_j)$ be the total number of conflicts between Q_i and Q_j, that is

$$c(Q_i, Q_j) = \sum_e \frac{1 + c_n(e)}{2},$$

where the sum is taken over all edges between Q_i, Q_j. By the global stability of $\mathcal{P}$, the partition

$$\mathcal{P}_i = Q_i \cup \left(\bigcup_{j:j\neq i} Q_j\right)$$

gives not less conflicts. So we see that

$$c(Q_i, \bigcup_{j:j\neq i} Q_j) = \sum_{j:j\neq i} c(Q_i, Q_j) \geq 0$$

for all i. Summing these inequalities we obtain

$$(\ell - 1)\sum_{i<j} c(Q_i, Q_j) \geq 0.$$

Thus the partition $\bigcup_{i=1}^{\ell} Q_i$ gives not less conflicts than $\mathcal{P}$, and the statement follows. □

Lemma 1. *Complete bipartite graphs $K_{m,m}$ and $K_{m,m+1}$ are globally stable.*

Proof. Let U, V be the vertex sets of $K_{m,m}$. (That is, all edges of $K_{m,m}$ run between U and V.) Let $A \cup B$ be a partition of the vertex set $U \cup V$. Put $x = |A \cap U|$, $y = |A \cap V|$. Then

$$m - x = |B \cap U|, \quad m - y = |B \cap V|.$$

The number of edges between A and B is

$$x(m-y)+y(m-x)=m(x+y)-2xy.$$

This is not less than

$$|A|\cdot|B|/2=(x+y)(2m-x-y)/2.$$

For $K_{m,m+1}$, the computation is similar. With the same notation, the number of edges between A and B is

$$x(m+1-y)+y(m-x)=m(x+y)+x-2xy.$$

This is not less than

$$|A|\cdot|B|/2=(x+y)(2m+1-x-y)/2$$

since

$$x-2xy-(x+y)(1-x-y)/2=(x-y)(x-y+1)/2\geq 0$$

holds for $x,y\in\mathbb{Z}$. This implies the statement. □

Remark 1. *In the proof of Theorem 1 we only used the fact that $G(\mathcal{P}_j(n))$ is locally stable. However, the second inequality is the best possible in the sense that the constant 2 cannot be chosen smaller. Indeed, consider the natural embedding*

$$K_{m,m}\subset K_{m,m+1}\subset K_{m+1,m+1}\subset\ldots,$$

and consider $K_{m,n}$ to be a subgraph of K_{m+n}. Define the labeling c_{m+n} by

$$c_{m+n}(e)=\begin{cases}1, & \text{if } e\in E(K_{m,n}),\\ -1, & \text{if } e\in E(K_{m+n})\setminus E(K_{m,n}).\end{cases}$$

Then this labeling c_k is consistent and asymptotically $k^2/4$ edges have label 1. This gives an example that $g=1/2$, $\rho_1=1$ and $\rho_j=0$ for $j>1$, which attains the equality for the second inequality of Theorem 1.

III. Correlation clustering of positive integers with respect to coprimality

As we have mentioned already in the introduction, it is obvious that the role of the relation $\sim$ (or, in the graph model, the definitions of the labels ±1) is crucial for the structure of the optimal clustering. This motivates the investigations in the present section. We mention that the results presented here were published in the paper [1].

Consider the sequence of complete graphs K_n together with a hereditary labeling $c_n : E(K_n)\mapsto\{-1,1\}$. Labeling the vertices of K_n by the integers $A_n=\{1,\ldots,n\}$ the mapping c_n implies a reflexive and symmetric relation $\sim_n$ on A_n. By the hereditary property of c_n we may assume that $\sim_n$ admits this property, too, i.e., the restriction of $\sim_{n+1}$ to A_n is equal to $\sim_n$.

Let, more generally, $A_n\subset\mathbb{N}$ be finite and satisfying $A_n\subseteq A_{n+1}$ for $n=1,2,\ldots$. Assume that there is a reflexive and symmetric relation $\sim_n$ on A_n. Further assume that the restriction of $\sim_{n+1}$ to A_n is equal to $\sim_n$. Setting $A=\cap_{n=1}^{\infty}A_n$ we have $A\subseteq\mathbb{N}$. Define the relation $\sim$ on A as follows: for $a,b\in A$ we set $a\sim b$ if there exist $n\geq 1$ such that $a,b\in A_n$ and $a\sim_n b$. By the hereditary property of $\sim_n$ the relation $\sim$ is well defined on A, moreover it is reflexive and symmetric. Of course we can consider $\sim$ on $\mathbb{N}$, too. This justifies that in the sequel we consider a relation on $\mathbb{N}$. Divisibility is the best understood relation of integers. As it is not symmetric ($2|4$ but $4\nmid 2$) we cannot use it in our investigations. Fortunately the coprime relation, i.e. $a\sim b$ if $\gcd(a,b)>1$ or if $a=b=1$, is closely related to divisibility and is symmetric.

In this section we work with sets A_n of positive integers greater than 1, but not exceeding n (for $n=2,3,\ldots$). Moreover we assume that A_n is equipped with the above defined coprime relation. Note that the behavior of the gcd among the first n positive integers has been investigated from many aspects; see e.g. a paper of Nymann, [13].

Bakó and Aszalós [2] have made several experiments concerning the optimal clustering of A_n with respect to $\sim$. They have discovered that the classes of a near optimal clustering have regular structure. In the sequel denote by p_i the i-th prime, i.e., $p_1=2,p_2=3,\ldots$. Set

$$A_{i,n}=\{m\ :\ m\leq n,\ p_i|m,\ p_j\nmid n\ (j<i)\}.$$

In other words, $A_{i,n}$ is the set of integers at most n, which are divisible by p_i, but coprime to the smaller primes. Aszalós and Bakó found that

$$[2,n]\cap\mathbb{Z}=\bigcup_{j=1}^{\infty}A_{j,n} \qquad (2)$$

is an optimal correlation clustering for $n\leq 20$ and very probably for $n\leq 500$, too. Notice that $A_{j,m}=\emptyset$ for all large enough j, i.e., the union on the right hand side is actually finite.

The main result of this section (and of [1]) is that for

$$n_0=3\cdot5\cdot7\cdot11\cdot13\cdot17\cdot19\cdot23=111\ 546\ 435$$

the decomposition (2) is not optimal. We prove that the number of conflicts in

$$[2,n_0]\cap\mathbb{Z}=(A_{1,n_0}\cup\{n_0\})\cup(A_{2,n_0}\setminus\{n_0\})\bigcup_{j=3}^{\infty}A_{j,n} \qquad (3)$$

is less than in (2) with $n=n_0$. We cannot prove that (2) is optimal for $n<n_0$. However, we show that the natural greedy algorithm (Algorithm 1), presented below, produces the clustering (2) for all $n<n_0$, and it yields (3) for $n=n_0$.

As we have mentioned already in the introduction, by results from [3] we know that to find an optimal correlational clustering is an NP-hard problem. Hence to find an approximation of the optimal solution, it is natural to use some kind of greedy algorithm. For the sets A_n we use the following approach. The optimal clustering for $A_2=\{2\}$ is itself. Assume that we have a partition of A_{n-1} with $n>2$, and adjoin n to that class, which causes the less new conflicts. As a result we obtain a locally optimal clustering, which is not necessarily globally optimal on A_n.

Starting with a partition of A_{n-1} this algorithm returns a partition of A_n such that the conflicts caused by putting n into one of the classes is minimal. The output of Algorithm 1 on

Algorithm 1 Natural greedy algorithm

```
Require: an integer $n \geq 2$
Ensure: a partition $\mathcal{P}$ of $N$
$\mathcal{P} \leftarrow \{\{2\}\}$;
if $n = 2$ then return $\mathcal{P}$
end if
$m \leftarrow 3$
while $m \leq n$ do
    $\mathcal{P}_M \leftarrow \mathcal{P} \cup \{\{m\}\}$
    $M \leftarrow$ CONFLICTS($\mathcal{P}_M, m$)    ▷ the number of conflicts with respect to the partition $\mathcal{P}_M$ caused by the pairs $(m, a),\ a < m$
    $C \leftarrow$ number of classes in $\mathcal{P}$
    $j \leftarrow 1$
    while $j \leq C$ do
        $O \leftarrow$ OP($j, \mathcal{P}$)    ▷ $OP(j, \mathcal{P})$ denotes the $j$-th class in the partition $\mathcal{P}$.
        $\mathcal{P}_1 \leftarrow \mathcal{P} \setminus \{O\}$
        $\mathcal{P}_1 \leftarrow \mathcal{P}_1 \cup \{O \cup \{m\}\}$
        $M_1 \leftarrow$ NUPAIR($\mathcal{P}_1, m$)    ▷ the number of pairs $(m, a)$ with $a < m$ causing a conflict in the partition $\mathcal{P}_1$
        if $M_1 < M$ then
            $M \leftarrow M_1$
            $\mathcal{P}_M \leftarrow \mathcal{P}_1$
        end if
    end while
end while
return $\mathcal{P}_M$
```

the input n is denoted by $G(n)$. It is certainly a clustering of A_n. As one can easily check, we obtain

$$\begin{aligned}
G(3) &= \{\{2\},\{3\}\} \\
G(4) &= \{\{2,4\},\{3\}\} \\
G(5) &= \{\{2,4\},\{3\},\{5\}\} \\
G(6) &= \{\{2,4,6\},\{3\},\{5\}\} \\
&\vdots \\
G(15) &= \{\{2,4,6,8,10,12,14\},\{3,9,15\}, \\
&\quad \{5\},\{7\},\{11\},\{13\}\}.
\end{aligned}$$

For these values of n one can readily show that the above partitions provide (in fact the unique) optimal clusterings for A_n $(2 \leq n \leq 15)$, as well.

The main result of this section is the following

Theorem 3. *If $m < n_0 = 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19 \cdot 23 = 111\ 546\ 435$ then*

$$G(m) = \bigcup_{j=1}^{\infty} A_{j,m} \tag{4}$$

holds. However, we have

$$G(n_0) = (A_{1,n_0} \cup \{n_0\}) \cup (A_{2,n_0} \setminus \{n_0\}) \bigcup_{j=3}^{\infty} S_{j,n_0}.$$

Elements of the proof of Theorem 3. Since the complete proof of Theorem 3 is given in [1], here we only indicate the main ingredients of the proof. On this way, we recall several lemmas from [1], always without proofs.

The first important information we need is to characterize that class of $G(n-1)$ to which Algorithm 1 adjoins n. This is done with the following

Lemma 2. *Let $n > 2$ be an integer. Write $G(n-1) = \{P_1, \ldots, P_M\}$ and set $P_0 = \emptyset$. For $1 \leq j \leq M$ let*

$$E_{j,n} = \{m \ :\ m \in P_j, \gcd(m,n) = 1\}$$

and

$$B_{j,n} = \{m \ :\ m \in P_j, \gcd(m,n) > 1\}.$$

Define $E_{0,n} = B_{0,n} = \emptyset$. Let J be the smallest index for which $|B_{j,n}| - |E_{j,n}|$ $(j = 0, \ldots, M)$ is maximal. Then $G(n) = \{P'_0, \ldots, P'_M\}$ such that

$$P'_j = \begin{cases} P_j \cup \{n\}, & \text{if } j = J, \\ P_j, & \text{otherwise}. \end{cases}$$

This lemma has the following important consequence.

Corollary 3. *The following assertions are true.*

(1) *If n is even, then $n \in A_{1,n}$.*
(2) *If n is a prime, then $\{n\} \in G(n)$.*
(3) *If the smallest prime factor of n is p_i and $n \in S_{j,n}$, then $j \leq i$.*

The next result gives a useful bound for the sizes of the sets $A_{i,u}$.

Lemma 3. *Let u be an odd integer. Then we have $|A_{1,u}| = \frac{u-1}{2}$. Further, if p_i is an odd prime, then*

$$\left| |A_{i,u}| - \frac{u}{p_i} \prod_{\ell=1}^{i-1} \left(1 - \frac{1}{p_\ell}\right) \right| \leq 2^{i-2}.$$

The next lemma provides an estimation for $|B_{j,n}| - |E_{j,n}|$, where

$$B_{j,n} = \{m \ :\ m \in A_{j,n-1},\ \gcd(m,n) > 1\}$$

and

$$E_{j,n} = \{m \ :\ m \in A_{j,n-1},\ \gcd(m,n) = 1\}.$$

Note that the elements of $B_{j,n}$ and $E_{j,n}$ are those elements of $A_{j,n-1}$, which are, and which are not in the relation $\sim$ with n, respectively.

Lemma 4. *Let $q_1 < \cdots < q_t$ be odd primes, $\alpha_1, \ldots, \alpha_t$ positive integers and $n = q_1^{\alpha_1} \cdots q_t^{\alpha_t}$. Let $j \geq 2$ be such that $p_j < q_1$. Then*

$$\left| |B_{j,n}| - |E_{j,n}| - C_{n,j,t} \right| \leq 2^{t+j-2}$$

holds, where

$$C_{n,j,t} = \frac{n-1}{p_j} \prod_{\ell=1}^{j-1} \left(1 - \frac{1}{p_\ell}\right) \left(1 - 2 \prod_{k=1}^{t} \left(1 - \frac{1}{q_k}\right)\right).$$

The next lemma plays an important role in the proof of Theorem 3. We use the previous notation.

Lemma 5. *Let $n = q_1^{\alpha_1} \cdots q_t^{\alpha_t}$ with $q_1 < \cdots < q_t$ odd primes and $\alpha_1, \dots, \alpha_t$ positive integers. Then*

$$\begin{aligned} |E_{1,n}| &= \frac{\varphi(n)}{2} = \frac{n}{2}\left(1 - \frac{1}{q_1}\right) \cdots \left(1 - \frac{1}{q_t}\right), \\ |B_{1,n}| &= \frac{n-1}{2} - |E_{1,n}|. \end{aligned}$$

Now, in principle, we are ready to give the main steps of the proof of Theorem 3. However, the proof is rather detailed, tricky and complicated, so we restrict ourselves to indicate how the argument proceeds. We refer the interested reader again to [1] for details.

Step 1. We start with confirming the cases where n is odd and $3 \mid n$. The difficult part is to show that (4) holds for $n < n_0$. This assertion is verified by comparing the estimates of Lemmas 2, 3, 4, and 5, some computer search, and applying a tool from prime number theory namely, estimates for expressions of the form

$$\prod_{p<x} \left(1 - \frac{1}{p}\right).$$

For the latter one can use e.g. formulas from [14].

Step 2. Next we check Theorem 3 for integers n with one or two prime factors. For this we need to combine Lemmas 2, 3, and 4, involved computer search, and the following two lemmas are also needed. The first one verifies Theorem 3 if n is a prime power.

Lemma 6. *Let $p = p_i \geq 3$ be a prime. If $p \leq 67$ and $p^\alpha < n_0$, $\alpha > 0$ then $p^\alpha \in A_{i,n}$. In general, $n = p^\alpha \in A_{i,n}$ holds for $\alpha \leq 4$.*

The second lemma proves our theorem for n with two distinct prime divisors, where the smaller one is at most 53.

Lemma 7. *Let $p = p_i \geq 3$ and $q > p$ be primes. If $p \leq 53$ and $p^\alpha q^\beta < n_0, \alpha, \beta > 0$ then $p^\alpha q^\beta \in A_{i,n}$. In general, $n = pq \in A_{i,n}$ is valid whenever $q < p^3$.*

Step 3. Consider now numbers n with three distinct prime factors. Unfortunately, for such values of n we could not find any general assertion or formula like in Lemma 6 or 7. However, our previous calculations yielded that here we may assume that the smallest prime factor of n is at least 19. For each prime $29 \leq p \leq 43$ we computed all integers, which are divisible by p, lie below a preliminary computed bound, and have three distinct prime divisors, which are at least p. Then we used a variant of the wheel algorithm, see e.g. [17], to handle these cases. Altogether, up to this point we could cover all values of n whose smallest prime factor is at most 47.

Step 4. To cover the remaining values of n (which have only "large" prime factors), we applied again formulas concerning the distribution of primes. Namely, we used estimates for $\pi(x)$, from [14]. This finishes the proof of Theorem 3. □

We proved that applying Algorithm 1 for A_n $(n \geq 1)$, the outputs (i.e. the clusterings of the A_n) have a regular shape until a certain large value of n (in fact up to n_0), but at that point the regularity vanishes. From the proof it is clear that n_0 is the first, but not at all the last integer, which behaves in this irregular way. For example, the number $3n_0$ is odd and is divisible by 3, however, adjoining it to $A_{1,n}$ causes less conflicts than adjoining it to $A_{2,n}$. Let $A_{i,n}^*$ denote the class containing p_i, produced by Algorithm 1. We can neither guess the structure of these sets, nor what is their asymptotic behavior. For example, it would be interesting to know whether the limit

$$\lim_{n\to\infty} \frac{|A_{1,n}^*|}{n}$$

exists or not, or is

$$\limsup_{n\to\infty} \frac{|A_{1,n}^*|}{n} = 1$$

or not.

On the other hand Theorem 1 shade some light to the asymptotic behavior of the optimal correlation clustering of $(A_n, \sim)$. Indeed denote by $\underline{g}, g, \overline{g}$, and $\overline{\rho_j}, j \geq 1$ the quantities defined in Section II in the case $(A_n, \sim)$. The next theorem is new.

Theorem 4. *With the above notation we have*

$$\overline{\rho_1} \leq \sqrt{2\left(1 - \frac{6}{\pi^2}\right)} = 0.885520071...$$

In particular the optimal correlation clustering of $(A_n, \sim)$ has at least two classes.

Proof. First we prove that in the actual case g exists, i.e., $\underline{g} = g = \overline{g}$. Indeed we have

$$g = \lim_{n\to\infty} \frac{a_n}{b_n},$$

where

$$a_n = |\{(a,b) \ : \ 1 \leq a \leq b \leq n, \gcd(a,b) > 1\}|,$$

and

$$b_n = |\{(a,b) \ : \ 1 \leq a \leq b \leq n\}| = \frac{n(n-1)}{2}.$$

Obviously

$$\begin{aligned} a_n &= b_n - |\{(a,b) \ : \ 1 \leq a \leq b \leq n, \gcd(a,b) = 1\}| \\ &= b_n - \sum_{d=1}^{n} \varphi(d), \end{aligned}$$

where $\varphi(x)$ denotes Euler's totient function. It is well known, see e.g. [10], that

$$\sum_{d=1}^{n} \varphi(d) = \frac{3}{\pi^2} n^2 + O(n \log n).$$

Combining everything together we get

$$g = 1 - \frac{6}{\pi^2}.$$

By the second assertion of Theorem 1 we get

$$\underline{\rho}_1^2 \leq 2g,$$

which together with the last inequality implies the statement of the theorem. □

IV. CORRELATION CLUSTERING OF S-UNITS WITH RESPECT TO COPRIMALITY

In this section we perform a similar analysis as in Section III, with the same relation, but the set of positive integers replaced by the set of positive integers having all prime divisors in a preliminary fixed finite set. As it will turn out, this modification changes the structure of the optimal clustering drastically. All the results presented in this section are new.

To formulate our results in this direction, we need to introduce some notions and notation. In what follows, let $S = \{p_1, \dots, p_k\}$ be a finite set of primes. Those positive integers which has no prime divisors outside S will be called S-units, and their set is denoted by $\mathbb{Z}_S$. This terminology is widely used in number theory, but in computer science the elements of the set $\mathbb{Z}_S$ for $S = \{2, 3, 5\}$ are also called Hamming numbers, see e.g. [8]. For a given positive $x \in \mathbb{R}$, let $\mathbb{Z}_S(x)$ denote the subset of $\mathbb{Z}_S$ consisting of S-units not greater than x. The sets $\mathbb{Z}_S(n), n = 1, 2, \dots$ play the same role as A_n in the last section. First we give a sharp upper bound for the number of elements of $\mathbb{Z}_S(x)$ and some of its subsets. For this we need some preparation. The following result due to Davenport [7] will be very useful.

Lemma 8 ([7, Theorem]). *Let $\mathcal{R}$ be a closed bounded region in the n dimensional space $\mathbb{R}^n$ and let $\mathrm{N}(\mathcal{R})$ and $\mathrm{V}(\mathcal{R})$ denote the number of points with integral coordinates in $\mathcal{R}$ and the volume of $\mathcal{R}$, respectively. Suppose that:*

- *Any line parallel to one of the n coordinate axes intersects $\mathcal{R}$ in a set of points which, if not empty, consists of at most h intervals.*
- *The same is true (with m in place of n) for any of the m dimensional regions obtained by projecting $\mathcal{R}$ on one of the coordinate spaces defined by equating a selection of $n - m$ of the coordinates to zero; and this condition is satisfied for all m from 1 to $n - 1$.*

Then

$$|\mathrm{N}(\mathcal{R}) - \mathrm{V}(\mathcal{R})| \le \sum_{m=0}^{n-1} h^{n-m} V_m,$$

where V_m is the sum of the m dimensional volumes of the projections of $\mathcal{R}$ on the various coordinate spaces obtained by equating any $n - m$ coordinates to zero, and $V_0 = 1$ by convention.

The next lemma will also play an important role later on.

Lemma 9. *Let $y_1, \dots, y_r, x$ be positive real numbers, and let $N(\mathbf{y}, x)$ denote the number of non-negative integer solutions $n_1, \dots, n_r$ of the inequality*

$$0 \le y_1 n_1 + \dots + y_r n_r \le x. \tag{5}$$

Then we have

$$N(\mathbf{y}, x) = c(\mathbf{y}) x^r + O(x^{r-1}),$$

where $c(\mathbf{y})$ is the volume of the r-dimensional polyhedron defined by the inequalities

$$\begin{aligned} x_i &\ge 0 \ (i = 1, \dots, r), \\ y_1 x_1 + \dots + y_r x_r &\le 1. \end{aligned}$$

Proof. It is clear that the non-negative integers $n_1, \dots, n_r$ satisfy (5) if and only if the lattice point $(n_1, \dots, n_r)$ belongs to the polyhedron $P(\mathbf{y}, x)$ defined by the inequalities

$$\begin{aligned} x_i &\ge 0 \ (i = 1, \dots, r), \\ y_1 x_1 + \dots + y_r x_r &\le x. \end{aligned}$$

Hence it is sufficient to bound the number of lattice points inside $P(\mathbf{y}, x)$. Obviously, $P(\mathbf{y}, x)$ satisfies the conditions of Lemma 8. Moreover, the volume of $P(\mathbf{y}, x)$ equals x^r times the volume of $P(\mathbf{y}, 1)$. The domains V_m occurring in Lemma 8 are polyhedra of dimensions at most $r - 1$, hence their total volume can be bounded by $O(x^{r-1})$, and the statement follows. □

Now using Lemma 9 we can easily bound the number of elements of $\mathbb{Z}_S(x)$.

Corollary 4. *Letting $c_S = c(\log p_1, \dots, \log p_k, 1)$, we have*

$$|\mathbb{Z}_S(x)| = c_S (\log x)^k + O((\log x)^{k-1}).$$

Proof. A positive integer m belongs to $\mathbb{Z}_S(x)$ if and only if $m \le x$ and there exist non-negative integers $n_1, \dots, n_k$ such that

$$m = p_1^{n_1} \cdots p_k^{n_k}.$$

This implies that

$$0 \le \log m = n_1 \log(p_1) + \dots + n_k \log(p_k) \le \log x.$$

Since $\log p_1, \dots, \log p_k > 0$ are real numbers, the conditions of Lemma 9 are satisfied, and the statement follows. □

Now we shall investigate correlation clustering on $\mathbb{Z}_S$ equipped with the same coprimality relation which we used in Section III. More precisely, for $a, b \in \mathbb{Z}_S$ let $a \sim b$, if and only if $\gcd(a, b) > 1$ or $a = b = 1$. For an integer $n \ge 1$ let $\mathcal{P}$ be a partition of $\mathbb{Z}_S(n)$. As before, we say that the S-units a and b are in conflict with respect to $\mathcal{P}$, if either they are in the same class but $a \not\sim b$, or they are in different classes and still $a \sim b$. As before, the purpose of correlation clustering is to find a partition with minimal number of conflicts.

The partition of $\mathbb{Z}_S(n)$ with only one class (i.e. when all elements of $\mathbb{Z}_S(n)$ belong to the same class) is called the trivial partition.

Lemma 10. *Let $C_S(n)$ denote the number of conflicts in the trivial partition of $\mathbb{Z}_S(n)$. Then we have*

$$C_S(n) \le c(\log n)^k,$$

where c is a positive constant.

Proof. In case of the trivial partition, two different S-units are in conflict precisely when they are coprime. Thus we have to count those pairs of S-units (a, b), $a \ne b$ which are coprime. Let a, b be distinct S-units (not necessarily elements of $\mathbb{Z}_S(n)$), and assume that $\gcd(a, b) = 1$. Then there exists a $T \subseteq S$, such that

$$a = \prod_{p_j \in T} p_j^{\alpha_j}, \quad b = \prod_{p_j \in S \setminus T} p_j^{\alpha_j}.$$

If $T=\emptyset$ or $S\setminus T=\emptyset$ then the empty product is defined as 1, i.e. the pairs (a,b) with $a=1$ or $b=1$ are included. These equations mean that a is a T-unit, and b is an $(S\setminus T)$-unit. Hence for fixed $T\subseteq S$, the $(S\setminus T)$-units up to n are the positive integers which are coprime to the T-units in $\mathbb{Z}_S(n)$. Thus

$$C_S(n)\le\sum_{T\subseteq S}|\mathbb{Z}_T(n)|\cdot|\mathbb{Z}_{S\setminus T}(n)|.$$

Using Corollary 4 and $|T|+|S\setminus T|=k$, we obtain that

$$\begin{aligned}C_S(n)&\le\sum_{T\subseteq S}c_Tc_{S\setminus T}(\log n)^{|T|}(\log n)^{|S\setminus T|}+\\&+\;O((\log n)^{|T|+|S\setminus T|-1}=\\&=\left(\sum_{T\subseteq S}c_Tc_{S\setminus T}\right)(\log n)^k+O((\log n)^{k-1}).\end{aligned}$$

Clearly, for fixed S the sum

$$\sum_{T\subseteq S}c_Tc_{S\setminus T}$$

is independent of n, hence it is a constant. This proves the statement. □

Denoting by $g=g(\mathbb{Z}_S)$ and $\underline{\rho}_1=\underline{\rho}_1(\mathbb{Z}_S)$ the densities defined in Section II, in the actual case we obtain

Corollary 5. *We have*

$$g=\underline{\rho}_1=1.$$

Proof. Corollary 4 and Lemma 10 imply $g=1$ immediately. Thus by Corollary 1 we have $\sum_j\underline{\rho}_j\ge1$, which together with the obvious inequality $\sum_j\underline{\rho}_j\le1$ implies that any optimal clustering of $\mathbb{Z}_S(n)$ is full. Hence we have $\underline{\rho}_1=1$ by Corollary 2. □

The equality $\underline{\rho}_1=1$ does not mean that the optimal correlation clustering is asymptotically trivial, i.e. the optimal correlation clustering of $\mathbb{Z}_S(n)$ is trivial for all large enough n. Thus the statement of the next theorem is much sharper as that of the last corollary.

Theorem 5. *Suppose that n is large enough. Then the optimal correlation clustering of $\mathbb{Z}_S(n)$ is given by the trivial partition.*

Proof. If $k=1$, i.e. S contains only one element, then all members of $\mathbb{Z}_S$ are divisible by it. Thus $\sim$ is an equivalence relation on $\mathbb{Z}_S$, which corresponds to the trivial partition. Hence the proof is complete for $k=1$ and we may assume $k\ge2$ in the sequel.

Let $1\le i\le k$ arbitrary, and let $a\in\mathbb{Z}_S(n)$ be such that $p_i\mid a$. Denote by Z_a the set of elements $b\in\mathbb{Z}_S(n)$ with $a\ne b$, $\gcd(a,b)>1$. It is clear that Z_a contains all elements of $\mathbb{Z}_S(n)\setminus\{a\}$, which are multiples of p_i. Thus the number of these elements can be bounded by the number of elements of the set obtained from $\mathbb{Z}_S(n)$ by omitting its elements which are not divisible by p_i. Observe that the latter set is just $\mathbb{Z}_{S\setminus\{p_i\}}(n)$. Thus by Corollary 4 we obtain

$$\begin{aligned}|Z_a|&\ge c_S(\log n)^k+O((\log n)^{k-1})-\\&-\;c_{S\setminus\{p_i\}}(\log n)^{k-1}+O((\log n)^{k-2})\qquad(6)\\&=c_S(\log n)^k+O((\log n)^{k-1}).\end{aligned}$$

Consider now an optimal clustering of $\mathbb{Z}_S(n)$, denoted by $\mathcal{P}$. In view of Lemma 10, the number of conflicts with respect to $\mathcal{P}$ is at most $c(\log n)^k$, where c is a positive constant.

Assume that the elements of Z_{p_i} are distributed in r different classes, which contain $m_1,\dots,m_r$ elements of Z_{p_i}, respectively. As the elements divisible by p_i corresponding to different classes are in conflict, thus the total number of conflicts with respect to $\mathcal{P}$ is at least

$$V=\sum_{1\le j_1<j_2\le r}m_{j_1}m_{j_2}.$$

Assume that

$$\max_{1\le j\le r}m_j\le(\log n)^{1/2}.$$

Then by (6) we have

$$r\ge c_S(\log n)^{k-1/2},$$

whence

$$V\ge\frac{r(r-1)}{2}>\frac{r^2}{4}\ge\frac{c_S^2}{4}(\log n)^{2k-1},$$

whose magnitude is larger than that of the conflicts of the trivial partition. This implies that for an optimal clustering

$$\max_{1\le j\le r}m_j>(\log n)^{1/2}$$

holds.

Let now $d>(c+1)/c_S$, where c is the constant given in Lemma 10. Suppose that

$$(\log n)^{1/2}<\max_{1\le j\le r}m_j<|Z_{p_i}|-d.$$

Assume further that the m_j take their maximum for $j=1$. Then

$$\begin{aligned}V\ge m_1(m_2+\dots+m_r)&=m_1(m_1+\dots+m_r-m_1)\\&=m_1(|Z_{p_i}|-m_1).\end{aligned}$$

The function $m_1(|Z_{p_i}|-m_1)$ can attain its minimum in the endpoints of the given interval. If $m_1=(\log n)^{1/2}$, then by (6) we have

$$V\ge c_S(\log n)^{k+1/2},$$

while $m_1=|Z_{p_i}|-d$ implies

$$V\ge d(|Z_{p_i}|-d)>(c+1)(\log n)^k+O((\log n)^{k-1}).$$

Hence we get that the number of conflicts in both endpoints are larger than that of the trivial partition.

This implies that there exists a class, say $\mathcal{P}_1$, in which the number of multiples of p_i is at least $|Z_{p_i}|-d$. Suppose that $d>0$ and let a be such that $p_i|a$, but $a\notin\mathcal{P}_1$. Then a is in conflict with all the elements of $\mathcal{P}_1\cap Z_{p_i}$, and the number of such elements is

$$|Z_{p_i}|-d=c_S(\log n)^k+O((\log n)^{k-1}).$$

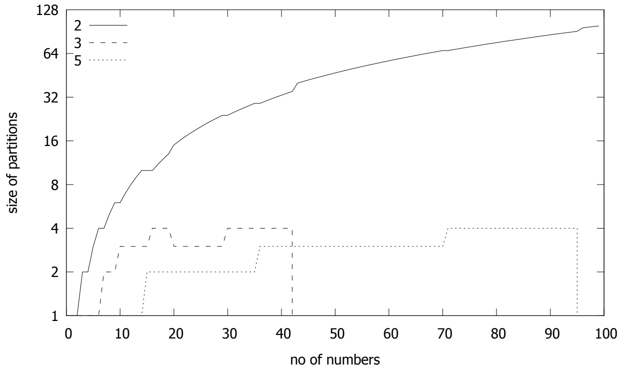


Fig. 1. Simulation result: from the 96^{th} Hamming numbers the correlation clustering gives only one cluster.

Moving a to $\mathcal{P}_1$, these conflicts will disappear. Of course, new conflicts can arise, on the one hand with those $d-1$ multiples of p_i which are not in $\mathcal{P}_1$, and on the other hand, with the elements not divisible by p_i. Altogether, the number of new conflicts can be at most

$$d + |\mathbb{Z}_{S\setminus\{p_i\}}(n)| = O((\log n)^{k-1}).$$

That is, if $\mathcal{P}$ is an optimal correlation clustering and if n is large enough, then all multiples of p_i must belong to the same class. Since p_i has been chosen arbitrarily, thus all elements must belong to the same class, and the theorem follows. □

Remark 2. *Theorem 5 describes completely the optimal correlation clustering of $\mathbb{Z}_S(n)$ only if n is large enough. Choosing S as the set of the first k primes, $\mathbb{Z}_S(n) = A_n$ if $n < p_{k+1}$, thus the optimal correlation clusterings of $\mathbb{Z}_S(n)$ and A_n are identical. By the results of Section III the number of clusters and their sizes of the optimal correlation clustering of $\mathbb{Z}_S(n)$ are growing for small n-s. After a certain point this tendency changes, all but one clusters become smaller until only one cluster survives, like a black hole. Our experiments show that this happens already for relatively small values of n, as you can see in Fig. 1 for the Hamming numbers.*

This means that locally optimal algorithms, like Algorithm 1, cannot give globally optimal solution for the correlation clustering problem.

V. CONCLUSION

In this paper we have considered the problem of correlation clustering, introduced in [3], from three different but closely related aspects. First we have derived new results for the graph model of the problem, considering an increasing family of graphs. We have obtained results concerning the optimal clustering in the general case. Then we have investigated particular sets with a specific relation. The reason for doing so is that clearly, the choice of the underlying relation strongly influences the structure of the optimal clustering. First we have considered positive integers and a relation based upon coprimality. Our main result here (recalled from [1]) has been that a natural greedy algorithm provides a "locally" optimal clustering up to a certain positive integer n_0, however, at some point the structure of such a clustering is deemed to change. Finally, we have considered the set of so-called S-units, under the same relation as for positive integers. Here we have proved that interestingly, in contrast with the case of positive integers, after some point the optimal clustering is always given by the trivial clustering (consisting of a single class).

ACKNOWLEDGMENT

We are grateful to the referees for their valuable remarks.

Our research was supported in part by the OTKA grants NK104208, NK101680, and K100339, by the Japanese Society for the Promotion of Science (JSPS), Grant in aid 24540012, and by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The project is implemented through the New Hungary Development Plan, cofinanced by the European Social Fund and the European Regional Development Fund.

REFERENCES

[1] L. Aszalós, L. Hajdu, and A. Pethő, *On a correlational clustering of integers*, arXiv:1404.0904 [math.NT].

[2] M. Bakó, and L. Aszalós, *Combinatorial optimization methods for correlation clustering*, In: Coping with complexity/D. Dumitrescu, Rodica Ioana Lung, Ligia Cremene, Casa Cartii de Stiinta, Cluj-Napoca, 2–12, 2011.

[3] N. Bansal, A. Blum, and S. Chawla, *Correlational clustering*, Machine Learning, **56** (2004), 89–113.

[4] A. Bhattacharya, and R. K. De, *Divisive Correlation Clustering Algorithm (DCCA) for grouping of genes: detecting varying patterns in expression profiles.* Bioinformatics **24** 11. (2008): 1359–1366.

[5] Y. Chen, S. Sanghavi, and H. Xu, *Clustering sparse graphs.* Advances in neural information processing systems. (2012) 2204–2212.

[6] Z. Chen, S. Yang, L. Li, and Z. Xie, *A clustering approximation mechanism based on data spatial correlation in wireless sensor networks*, Wireless Telecommunications Symposium (WTS), 2010.

[7] H. Davenport, *On a principle of Lipschitz*, J. London Math. Soc. **26**, (1951). 179–183. *Corrigendum* ibid. **39** (1964), 580.

[8] E.W. Dijkstra, *A discipline of programming.* Vol. 4. Englewood Cliffs: Prentice-Hall, 1976. 129-133.

[9] T. DuBois, J. Golbeck, J. Kleint, and A. Srinivasan, *Improving recommendation accuracy by clustering social networks with trust.* Recommender Systems & the Social Web *532* (2009): 1-8.

[10] L.K. Hua, *Introduction to Number Theory*, Springer-Verlag, 1982.

[11] P. Kanani, and A. McCallum, *Resource-bounded information gathering for correlation clustering* In Computational Learning Theory 07, Open Problems Track, COLT 2007, 625-627, 2007.

[12] Z. Néda, F. Răzvan, M. Ravasz, A. Libál, and G. Györgyi, *Phase transition in an optimal clusterization model* Physica A: Statistical Mechanics and its Applications, **362** (2):357–368, 2006.

[13] J. E. Nymann, *On the probability that k positive integers are relatively prime*, J. Number Theory **4** (1972), 469–473.

[14] J. B. Rosser, and L. Schoenfeld, *Approximate formulas for some functions of prime numbers*, Illinois J. Math. **6** (1962), 64–94.

[15] K. Sungwoong, S. Nowozin, P. Kohli, and D. Y. Chang, *Higher-Order Correlation Clustering for Image Segmentation*, In: Advances in Neural Information Processing Systems 24. J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira, and K.Q. Weinberger, Curran Associates, Inc. 1530–1538, 2011.

[16] B. Yang, W. K. Cheung, and J. Liu, *Community mining from signed social networks*, IEEE Transactions on Knowledge and Data Engineering **19** 10 (2007): 1333-1348.

[17] H. C. Williams, *Primality testing on a computer*, Ars Combin. **5** (1978), 127–185.

Shigeki Akiyama is a professor at Institute of Mathematics, University of Tsukuba, Japan. He got PhD degree in mathematics from Kobe University. His research interest is the interplay between number theory and ergodic theory. He thinks clustering looks quite similar to how we understand our nature by language. You can contact him: akiyama@math.tsukuba.ac.jp

László Aszalós is a senior lecturer and researcher at the University of Debrecen, member of the Computer Science Department since 1997. He received his PhD at 2002 at this university. His research interests belong to AI: automated theorem proving, multi-modal logics, optimization and rough set theory. Recently he examine the correlation clustering: its near-optimal solutions and applications in data mining.You can contact him: aszalos.laszlo@inf.unideb.hu.

Lajos Hajdu received his MSc in Mathematics from the Lajos Kossuth University, Hungary, in 1992. He obtained his PhD degree in Mathematics from the Lajos Kossuth University, Hungary, in 1998. He worked as a Post Doc researcher for the Mathematical Institute of Leiden University in 1999-2000. From 2000 he served as Assistant Lecturer, from 2003 as Assistant Professor and since 2012 he has been a Full Professor at the Institute of Mathematics, University of Debrecen. He is a member of the János Bolyai Mathematical Society, the Public Body of the Hungarian Academy of Sciences, and the Mathematical Committee of the Mathematical Division of the Hungarian Academy of Sciences. He has authored or co-authored 80 journal papers and 10 conference papers. His main interest lies in Diophantine number theory, in discrete tomography and in discrete mathematics with applications in digital image processing. His email address is hajdul@science.unideb.hu.

Attila Pethő is a professor of the Department of Computer Science, Faculty of Informatics, University of Debrecen, Hungary. He got PhD degree in mathematics from the Lajos Kossuth University. He is a corresponding member of the Hungarian Academy of Sciences. His research interest are number theory and cryptography. You can contact him: petho.attila@inf.unideb.hu.

Crowdsensing Based Public Transport Information Service in Smart Cities

Károly Farkas, Gábor Fehér, András Benczúr, and Csaba Sidló, *Member, IEEE*

***Abstract*—Thanks to the development of technology and the emergence of intelligent services smart cities promise to their inhabitants enhanced perception of city life. For example, a live timetable service of public transportation can increase the efficiency of travel planning substantially. However, its implementation in a traditional way requires the deployment of some costly sensing and tracking infrastructure. Mobile crowdsensing is an alternative, when the crowd of passengers and their mobile devices are used to gather data for almost free of charge.**

In this paper, we put the emphasis on the introduction of our crowdsensing based public transport information service, what we have been developing as a prototype smart city application. The front-end interface of this service is called TrafficInfo. It is a simple and easy-to-use Android application which visualizes real-time public transport information of the given city on Google Maps. The lively updates of transport schedule information relies on the automatic stop event detection of public transport vehicles. TrafficInfo is built upon our Extensible Messaging and Presence Protocol (XMPP) based communication framework what we designed to facilitate the development of crowd assisted smart city applications. The paper introduces shortly this framework, than describes TrafficInfo in detail together with the developed stop event detector.

***Index Terms*—Crowdsensing, Public transport, GTFS, Publish/subscribe, XMPP, Android, Smart cities**

I. Introduction

SERVICES offered by smart cities aim to support the everyday life of inhabitants. Unfortunately, the traditional way of introducing a new service usually implies a huge investment to deploy the necessary background infrastructure.

One of the most popular city services is public transportation. Maintaining and continuously improving such a service are imperative in modern cities. However, the implementation of even a simple feature which extends the basic service functions can be costly. For example, let's consider the replacement of static timetables with lively updated public transport information service. It requires the deployment of a vehicle tracking infrastructure consisting of among others GPS sensors, communication and back-end informatics systems and user interfaces, which can be an expensive investment.

An alternative approach to collect real-time tracking data is exploiting the power of the crowd via participatory sensing or often called mobile crowdsensing[1] [1], which does not call for such an investment. In this scenario (see Fig. 1), the passengers' mobile devices and their built-in sensors, or the passengers themselves via reporting incidents, are used to generate the monitoring data for vehicle tracking and send instant route information to the service provider in real-time. The service provider then aggregates, cleans, analyzes the data gathered, and derives and disseminates the lively updates. The sensing task is carried out by the built-in and ubiquitous sensors of the smartphones either in participatory or opportunistic way depending on whether the user is involved or not in data collection. Every traveler can contribute to this data harvesting task. Thus, passengers waiting for a ride can report the line number with a timestamp of every arriving public transport vehicle at a stop during the waiting period. On the other hand, onboard passengers can be used to gather and report actual position information of the moving vehicle and detect halt events at the stops.

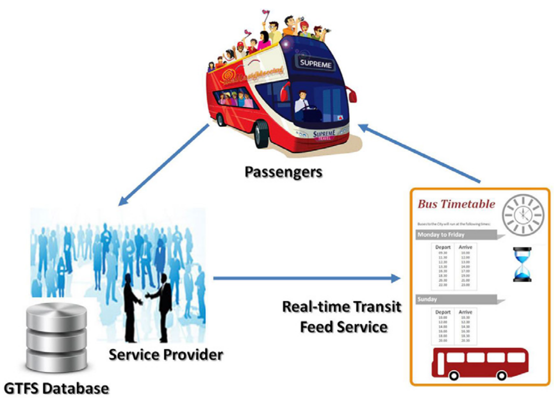


Fig. 1. Real-time public transport information service based on mobile crowdsensing

In this paper, we focus on the introduction of our crowdsensing based public transport information service, what we have been developing as a prototype smart city application. The front-end interface of this service, called TrafficInfo, is a simple and easy-to-use Android application which visualizes real-time public transport information of the given city on Google Maps. It is built upon our Extensible Messaging and Presence Protocol (XMPP) [2] based communication framework [3]

Manuscript submitted on October 19, 2014, revised on December 8, 2014.

K. Farkas is with the Inter-University Centre for Telecommunications and Informatics, Debrecen, Hungary and the Budapest University of Technology and Economics, Budapest, Hungary (corresponding author, e-mail: farkask@hit.bme.hu).

G. Fehér is with the Inter-University Centre for Telecommunications and Informatics, Debrecen, Hungary and the Budapest University of Technology and Economics, Budapest, Hungary (e-mail: feher@tmit.bme.hu).

A. Benczúr and Cs. Sidló are with the University of Debrecen, Hungary and the Institute for Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI), Budapest, Hungary (e-mail: benczur@info.ilab.sztaki.hu, sidlo@sztaki.hu).

[1] We use the terms *crowdsensing*, *crowdsourcing* and *participatory sensing* interchangeably in this paper.

what we designed to facilitate the development of crowd assisted smart city applications (we also introduce shortly this framework in Sec. III). Following the publish/subscribe (pub/sub) communication model the passengers subscribe in TrafficInfo, according to their interest, to traffic information channels dedicated to different public transport lines or stops. Hence, they are informed about the live public transport situation, such as the actual vehicle positions, deviation from the static timetable, crowdedness information, etc.

To motivate user participation in data collection we offer a day zero service to the passengers, which is a static public transportation timetable. It is built on the General Transit Feed Specification (GTFS, designed by Google) [4] based transit schedule data and provided by public transport operators. TrafficInfo basically presents this static timetable information to the users which is updated in real-time, if appropriate crowdsensed data is available. To this end, the application collects position data; the timestamped halt events of the public transport vehicles at the stops (our automatic detector is described in Sec. IV-D); and/or simple annotation data entered by the user, such as reports on crowdedness or damaged seat/window/lamp/etc. After analyzing the data gathered live updates are generated and TrafficInfo refreshes the static information with them.

The rest of the paper is structured as follows. After a quick overview of related work in Sec. II we introduce shortly in Sec. III our generic framework to facilitate the development of crowdsourcing based services. In Sec. IV, we show our live public transport information service together with the developed stop event detector. Finally, in Sec. V we summarize our work with a short insight to our future plans.

II. Related Work

In this section, we discuss the challenge of attracting users to participate in crowdsensing and review the relevant works in the field of crowd assisted transit tracking systems.

A crowdsourcing based service has to acquire the necessary information from its users who are producers and consumers at the same time. Therefore it is essential for the service provider to attract users. However, we face a vicious circle here. The users are joining the service if they can benefit from it and at the same time they contribute to keep running the service which can persuade others also to join. But how can the users be attracted if the service is not able to provide the expected service level due to the lack of contributors? This also means that the service cannot be widely spread without offering a minimum service level and until it has a sufficiently large user base.

Moovit[2] is a similar application to TrafficInfo which is meant to be a live transit app on the market providing real-time information about public transportation. It faces the above mentioned problem in many countries. Moovit has been successful only in those cities where it has already a mass of users, just like in Paris, and not successful in cities where its user base is low, e.g., in Budapest. In order to create a sufficiently large user base Moovit provides, besides live data, schedule based public transportation information as a day zero service, too. The source of this information is the company who operates the public transportation network. The best practice is for providing such information is using GTFS [4]. According to the GTFS developer page, currently 263 public transportation companies provide transit feeds from all over the world. Moovit partially relies on GTFS and is available in 350 cities attracting more than 6.5 million users. We also adopted this solution in TrafficInfo.

Several other mobile crowdsensing based transit tracking ideas have been published recently. For instance, the authors in [5] propose a bus arrival time prediction system based on bus passengers' participatory sensing. The proposed system uses movement statuses, audio recordings and mobile cell-tower signals to identify the vehicle and its actual position. The authors in [6] propose a method for transit tracking using the collected data of the accelerometer and the GPS sensor on the users' smartphone. The authors in [7] use smartphone sensors data and machine learning techniques to detect motion type, e.g., traveling by train or by car. EasyTracker [8] provides a low cost solution for automatic real-time transit tracking and mapping based on GPS sensor data gathered from mobile phones which are placed in transit vehicles. It offers arrival time prediction, as well.

These approaches focus on the data (what to collect, how to collect, what to do with the data) to offer enriched services to the users. However, our focus is on how to introduce such enriched services incrementally, i.e., how can we create an architecture and service model, which allows incremental introduction of live updates from participatory users over static services that are available in competing approaches. Thus, our approach complements the above ones.

III. Framework for Crowdsensing Based Smart City Applications

In this section, we shortly describe our generic framework [3], which is based on the XMPP publish-subscribe architecture, to aid the development of crowdsensing based smart city applications. TrafficInfo is implemented on top of this framework.

A. Communication Model

XMPP [2] is an open technology for real-time communication using Extensible Markup Language (XML) [9] message format. XMPP allows sending of small information pieces from one entity to another in quasi real-time. It has several extensions, like multi-party messaging [10] or the notification service [11]. The latter realizes a publish/subscribe (pub/sub) communication model [12], where publications sent to a node are automatically multicast to the subscribers of that node. This pub/sub communication scheme fits well with most of the mobile crowdsensing based applications. In these applications, the users' mobile devices are used to collect data about the environment (publish) and the users consume the services updated on the basis of the collected data (subscribe).

Hence, we use XMPP and its generic publish/subscribe communication model in our framework to implement interactions. In this model, we define three roles, like *Producer*,

[2] http://www.moovitapp.com/

Consumer and *Service Provider* (see Fig. 2). These entities interact with each other via the core service, which consists of event based pub/sub nodes.

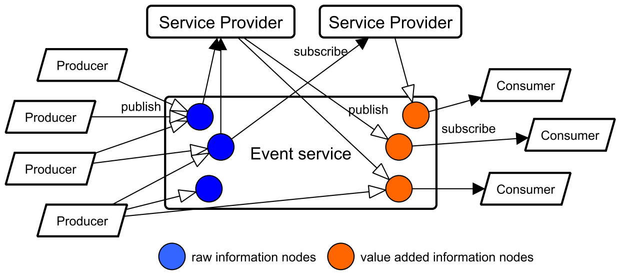


Fig. 2. Crowdsensing model based on publish/subscribe communication

Producer: The Producer acts as the original information source in our model producing raw data streams and plays a central role in data collection. He is the user who contributes his mobile's sensor data.

Consumer: The Consumer is the beneficiary of the provided service(s). He enjoys the value of the collected, cleaned, analyzed, extended and disseminated information. We call the user as *Prosumer*, when he acts in the service as both Consumer and Producer at the same time.

Service Provider: The Service Provider introduces added value to the raw data collected by the crowd. Thus, he intercepts and extends the information flow between Producers and Consumers. A Service Provider can play several roles at the same time, as he collects (Consumer role), stores and analyzes Producers' data to offer (Service Provider role) value added service.

In our model, depicted in Fig. 2, Producers are the source of original data by sensing and monitoring their environment. They publish (marked by arrows with empty arrowhead) the collected information to event nodes (raw information nodes are marked by blue dots). On the other hand, Service Providers intercept the collected data by subscribing (marked by arrows with black arrowhead) to raw event nodes and receiving information in an asynchronous manner. They extend the crowdsensed data with their own information or extract cleaned-up information from the raw data to introduce added value to Consumers. Moreover, they publish their service to different content nodes. Consumers who are interested in the reception of the added value/service just subscribe to the appropriate content node(s) and collect the published information also in an asynchronous manner.

B. Architecture

We can directly map this model to the XMPP publish/subscribe service model as follows (see Fig. 3):

- Service Providers establish raw pub/sub data nodes, which gather Producers' data, for the services they offer.
- Consumers can freely publish their collected data to the corresponding nodes with appropriate node access rights, too. However, only the owner or other affiliated Consumers can retrieve this information.

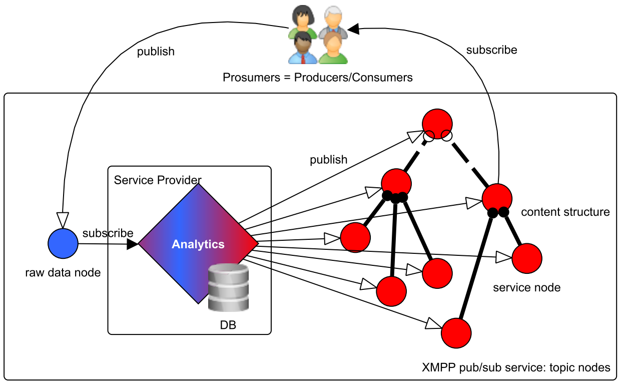


Fig. 3. Mobile crowdsensing: the publish/subscribe value chain using XMPP

- Producers can publish the collected data or their annotations to the raw data nodes at the XMPP server only if they have appropriate access rights.
- Service Providers collect the published data and introduce such a service structure for their added value via the pub/sub subscription service, which makes appropriate content filtering possible for their Consumers.
- Prosumers publish their sensor readings or annotations into and retrieve events from XMPP pub/sub nodes.
- Service Providers subscribed to raw pub/sub nodes collect, store, clean-up and analyze data and extract/derive new information introducing added value. This new information is published into pub/sub nodes on the other side following a suitable structure.

The pub/sub service node structure can benefit from the aggregation feature of XMPP via using collection nodes, where a collection node will see all the information received by its child nodes. Note, however, that the aggregation mechanism of an XMPP collection node is not appropriate to filter events. Hence, the Service Provider role has to be applied to implement scalable content aggregation. Fig. 3 shows XMPP aggregations as dark circles at the container node while empty circles with dashed lines represent only logical containment where intelligent aggregation is implemented through the service logic.

IV. Real-time Public Transport Information Service

In this section, we shortly overview the architecture of our public transport information service, then describe TrafficInfo, its front-end Android interface together with our stop event detector.

A. Service Architecture

Our real-time public transport information service architecture has two main building blocks, such as our crowdsensing framework described in Sec. III and the TrafficInfo application (see Fig. 4). The framework can be divided into two parts, a standard XMPP server and a GTFS Emulator with an analytics module.

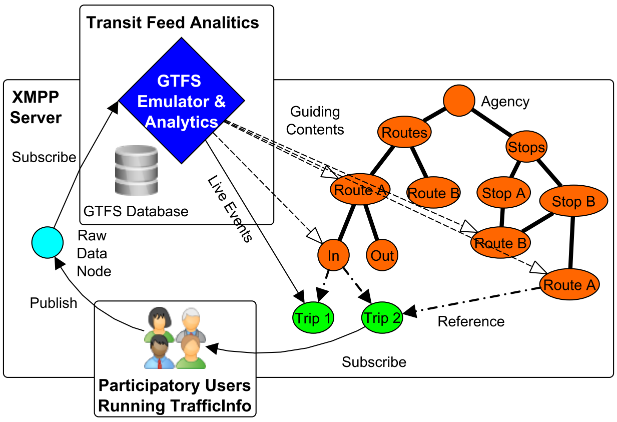


Fig. 4. Real-time public transport information service architecture

The XMPP server maps the public transport lines to a hierarchical pub/sub channel structure. We turned the GTFS database into an XMPP pub/sub node hierarchy. This node structure facilitates searching and selecting transit feeds according to user interest.

Transit information and real-time event updates are handled in the *Trip* nodes at the leaf level. The inner nodes in the node hierarchy contain only persistent data and references relevant to the trips. The users can access the transit data via two ways, based on *routes* or *stops*. When the user wants to see a given trip (vehicle) related traffic information the route based filtering is applied. On the other hand, when the forthcoming arrivals at a given stop (location) are of interest the stop based filtering is the appropriate access way.

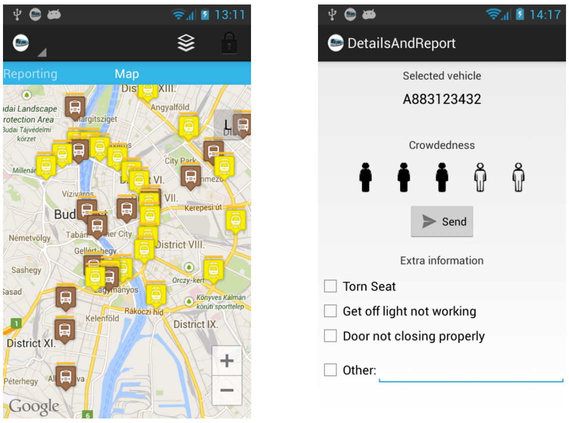
The GTFS Emulator provides the static timetable information, if it is available, as the initial service. It basically uses the officially distributed GTFS database of the public transport operator of the given city. However, it also relies on another data source, which is OpenStreetMap (OSM), a crowdsourcing based mapping service [13]. In OSM maps, users have the possibility to define terminals, public transportation stops or even public transportation routes. Thus, the OSM based information is used to extend and clean the information coming from the GTFS source. The analytics module is in charge of the business logic offered by the service, e.g., deriving crowdedness information or estimating the time of arrivals at the stops from the data collected by the crowd.

TrafficInfo handles the subscription to the pub/sub channels, collects sensor readings, publishes events to and receives updates from the XMPP server, and visualizes the received information.

B. TrafficInfo Features

TrafficInfo has three main features, but most of the users will benefit from its visualization capability that visualizes public transport vehicle movements on a city map.

1) Visualization: An example of this primary feature can be seen on Fig. 5a displaying trams 1, 4, 6 and buses 7 and 86 on the Budapest map in Hungary. The depicted vehicles can be filtered to given routes. The icon of a vehicle may reflect various attributes, such as the number, progress or crowdedness of the specific vehicle. Clicking on a vehicle's icon a popup shows all known information about that specific vehicle.



Selected vehicle

A883123432

Crowdedness

Send

Extra information

□ Torn Seat

□ Get off light not working

□ Door not closing properly

□ Other:

(a) Vehicle visualization

(b) User feedback form

Fig. 5. TrafficInfo screenshots

2) Information Sharing: The second feature is about information sharing. Passengers can share their observations regarding the vehicles they are currently riding. Fig. 5b shows the feedback screen that is used to submit the observations. The feedback information is spread out using our crowdsourcing framework and displayed on the devices of other passengers, who might be interested in it. It is up to the user what information and when he wants to submit, but we are planning to provide incentives to use this feature frequently.

3) Sensing: The third feature is collecting smartphone sensor readings without user interaction, which is almost invisible for the user. User positions are reported periodically and are used to determine the vehicle's position the passenger is actually traveling on. In order to create the link between the passenger and the vehicle, we try to identify the movement of the user through his activities. To this end we are using various sensors, e.g., accelerometer, and try to deduct the timestamped stop events of the vehicles (our automatic stop event detection mechanism is described in Sec. IV-D). The duration between the detected stops coupled with GPS coordinates identifies the route segment, which the user actually rides.

Besides the GPS coordinates Google also provides location information on those areas, where there is no GPS signal. Usually this position is highly inaccurate, but the estimated accuracy is also provided. We also use the activity sensor, which guesses the actual activity of the user. Currently, the supported activities are: *in vehicle*, *on bicycle*, *on foot*, *running*, *still*, *tilting*, *walking* and *unknown*. Accuracy is provided here, as well.

The collected sensor readings, on one hand, are uploaded to the XMPP server, where the analytics module processes and shares them among parties who are subscribers of the relevant information; on the other hand, are used locally. For example, user activity is analyzed on the server side and it is used to create non real-time stop patterns through machine

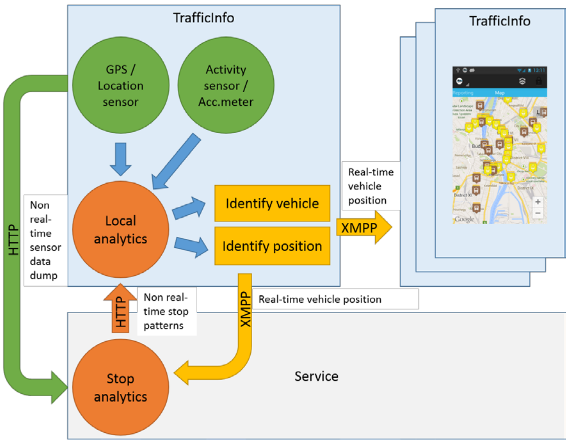

TrafficInfo
GPS / Location sensor
Activity sensor / Acc.meter
Local analytics
Identify vehicle
Identify position
Real-time vehicle position
XMPP
TrafficInfo
Non real-time sensor data dump
HTTP
Non real-time stop patterns
HTTP
XMPP
Real-time vehicle position
Service
Stop analytics

Fig. 6. Sensor data flows in TrafficInfo

learning. These patterns are delivered back to the application, where further local analytics can use them for, e.g., identifying the vehicle and providing position information. These sensor data flows are depicted in Fig. 6. Note that, at the moment, stop events are detected locally on the device due to resource usage reasons and only the detected events with a timestamp are reported back to the server. Based on this information the server side analytics estimate the upcoming arrival times of the given vehicle and disseminate live timetable updates to the subscribers.

C. Service Levels

Running TrafficInfo in a small city is different than in big cities, like Budapest. The cause of this difference is the unavailability of static public transportation information in, e.g., GTFS format. If even the static public transportation schedule is not presented by the application, people will likely not use it. Furthermore, fewer users will generate less live traffic data which makes the whole application useless. Hence, it is clear that we should apply a different approach in cities where static public transportation information has not been available, yet.

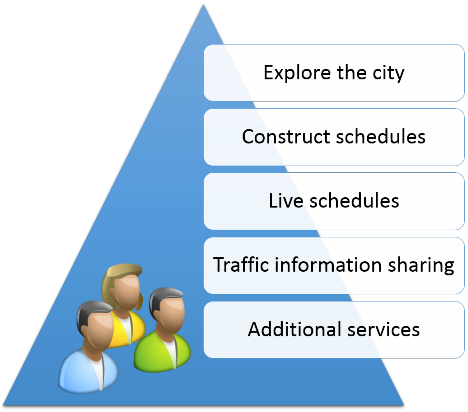


Fig. 7. Service levels vs. user base

Fig. 7 depicts the different service levels that we offer with the growing number of users. These service levels are the following.

1) City Explorer: In a new city, at the beginning we assume that there is zero knowledge in our system about the city's public transportation. The goal is to gather the relevant information in a fast and inexpensive way. When a reliable GTFS or OSM information base of the given city is available, we import this data into our databases. In other situations, we use crowdsourcing to gather this information. We assume that some users will install the TrafficInfo application either to contribute to city exploration or just simply for curiosity (or some incentive mechanism has to be introduced to grab users). We expect no other contributions than installing the application, carrying the smartphone during the day, traveling on public transportation and answering some simple questions asked by the application. The smartphones, using their built-in sensors, collect all the necessary data without user interaction. The questions are used to annotate the collected data.

Every day the captured data is uploaded in a batch to the server for analysis. At the same time, the application downloads information about what to ask on the following day(s).

Fig. 8 depicts an uploaded activity log of a particular user. In this example, the information source is the Google activity recognition module mentioned above. The blue bars show the detected activity during the capture time. In addition, another sensor module recorded the motion, too. Its output is the red bars, recognizing only still or moving states. The height of the bars expresses the confidence of the recognition. Althought the values represented with blue and red bars are coming from two different sensors, they usually have the same results for the still state. There are only a few differences, where the activity recogniton shows unknown event, while the motion sensor signals still state. It is not displayed on the figure, but the GPS position and its accuracy are also logged for every event.

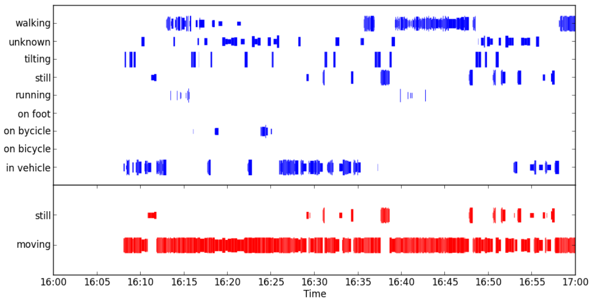


Fig. 8. Captured sensor data during user activity

The captured logs are processed by the server during the night, when the users are typically inactive and the system tries to guess the public transport stops and routes in the city. The more users report the same information the higher the chance is to guess the transportation system correctly. A database stores all the possible stop locations together with their confidence. This database is then downloaded to the application which will ask simple questions to the users to identify stops. For instance, the application might ask: *“Are you standing at a stop, waiting for public transport?”* We expect simple answers for simple questions until we can construct the public transportation stop database. Routes are explored in a similar way. When the user travels between already known stops, we assume that there is a public transport route among these stops. The application might ask the user about the route type and the line number.

2) Schedule (Re)Construction: Once the public transportation stops and routes are explored in most parts of the city, we can assume with high confidence that more users join and use the application. Visualizing stops and routes aids users to get orientation. However, the exploration of the city is continuing, the sensor readings are always collected, but questions are asked only regarding to the partially explored areas.

When the number of users exceeds a certain level and the trips can be guessed, the automatic detection of the stop events comes into the picture. The detected events are reported to the server by the application. The server filters this data and analyzes the patterns of each transport line. As more stop events are captured the patterns are more complete and finally the public transportation schedule is constructed.

3) Live Schedule: TrafficInfo providing public transportation stops, routes and schedules is assumed to attract many users, similarly to those applications that are available in big cities based on GTFS data. One advantage of TrafficInfo is that it provides an alternative way to collect all necessary information from scratch which does not require the cooperation of the public transport operator company, rather relies on the power of the crowd.

When the number of users is high enough and (static) schedule information is available, the continuously collected position and stop event data is used to create and propagate lively updates. These updates refresh the timetable if necessary and reflect the actual public transport traffic conditions.

4) Information Sharing on Public Transport Conditions: On-line users are able to send and receive information about the vehicle’s conditions they are actually riding. This requires user interaction on a voluntary basis as current sensors are not able to detect crowdedness, torn seats, bad drivers, etc. If the application has a wide user base we can always expect some volunteers to report on such conditions. The application provides easy to use forms to enter the relevant data (see Sec. IV-B2).

5) Additional Services: When TrafficInfo is running in a full-fledged manner, it can cooperate with other services targeting public transportation. For example, a rendezvous service can be paired to the TrafficInfo application to organize dates on public transportation vehicles.

D. Stop Event Detection

One of the fundamental functions of TrafficInfo is to detect stop events of public transport vehicles. We implemented such a detector locally on the mobile device. The reason behind that is twofold. First, cheaper devices produce bogus raw GPS location data that, if directly transmitted to the XMPP server, would mislead the service. Second, raw logs are generated at a very high rate and it would cause a substantial burden to transmit the raw logged data to the server in real-time for further processing. Instead, only when stop events are detected a summary of information, e.g., the timestamp of the event and the time elapsed since the last stop event, will be transmitted.

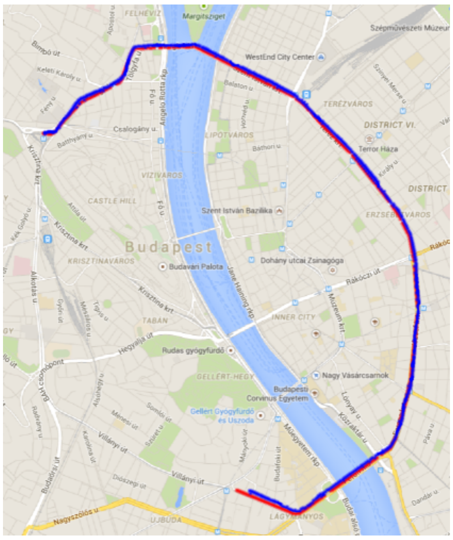


Fig. 9. GPS position trajectory (blue) and the real tram route (red) as logged by a Samsung Galaxy S3 device

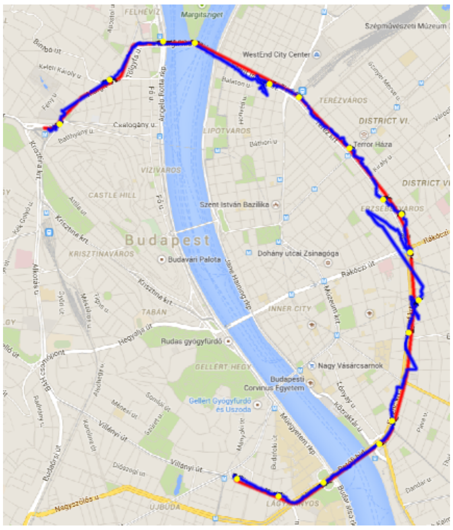


Fig. 10. GPS position trajectory (blue), the real tram route (red) and stops (yellow dots) as logged and detected by a Nexus4 device

To illustrate the challenge of stop event detection, we show the logged trajectory on tram routes 4 and 6 in Budapest from two devices, a Samsung Galaxy S3 and a Nexus4 smartphone, in Fig. 9 and Fig. 10, respectively. In case of Nexus4 (Fig. 10), yellow dots indicate the predicted locations of the stop events. Note that Nexus4 with network information provides correct position data, similar in quality to the Galaxy S3 device.

Unfortunately, we were not able to collect GPS position data from all the devices we used in our experiments even if the device was equipped with GPS sensor.

Our solution for stop event detection is based on features. Hence, we generated several features from the experimental usage logs collected during the testing period. The measurement object we used to collect context data is summarized in Table I. It includes among others GPS, WiFi, network and acceleration sensor readings, etc.

TABLE I
SEMANTICS OF TRAFFICINFO MEASUREMENT LOGS

Field Description	Examples, Possible Values
Event type	Initialization, manual, sensor
Timestamp	Time when the event occurred
Track (tram, bus line)	Tram 6
Phone type	E.g., Nexus4 including IMEI
Acceleration	Absolute or axes X, Y and Z
GSM signal strength	As defined in the relevant standard
Android GPS, network and passive location accuracy, longitude and Latitude values	Android GPS Location Provider data, accuracy radius with 68% confidence
CellID, WiFi MACID	LAC (Location Area Code) and CID (Cell ID)
Vehicle number	ID of the transport vehicle
Direction	Onward or backward
Arrived at	Time of arrival at the stop
Manual input	- Stopped at Station - Revoke Stopped at Station - Leaving Stop - Revoke Leaving Stop - Stopped at Traffic Light - Revoke Stopped at Traffic Light - Revoke Last Input

The features we defined are the following:

- Latitude, Longitude: raw GPS data;
- AccAbsMax and AccAbsMin: maximum and minimum value of acceleration in the past 20 seconds;
- Last Annotation Time: in seconds, depending on the annotation type (Stopped at Station or Leaving Stop);
- Closest Station: distance calculated from raw GPS data;
- GPS Distance: distance traveled during the last 20 seconds based on raw GPS data.

We collected Android sensor and location data by using the Android Location API[3]. The device can have multiple LocationProvider subclasses based on network, GPS and passive sensors, and the location manager has a method to select the best provider. Accessing the sensors requires three level permissions: ACCESS_FINE_LOCATION, ACCESS_COARSE_LOCATION, INTERNET. The GPS sensor can be accessed by the NMEA listener[4]. Accelerometer is accessible through the Google Location Services API, part of Google Play Services, a high level framework that automates the location provider choice.

For classification we used the J48 decision tree implementation of the Weka data mining tool[5]. The final output of our detector is the detected stop event, including location and timestamp. With the combination of the defined features and models we could detect stop events with high accuracy within 13 seconds after the arrival at the station.

We measure the accuracy of the method by computing the precision, recall and AUC (Area Under the Curve) [14] of our classifiers in a 10-fold crossvalidation setting. We consider AUC as the main stable measure for classifier performance that does not depend on the decision threshold separating the predicted stop events. The best classifier reached precision 0.97, recall 0.95, F measure 0.96. The corresponding best AUC was 0.86, which means that a random time point when the tram is at a stop is predicted 86% more likely a stop than another random time point when the tram is in between two stops. In general, an AUC between 0.8–0.9 is considered in the literature to be good to excellent.

V. SUMMARY

In this paper, we shortly introduced our XMPP based communication framework that we designed to facilitate the development of crowd assisted smart city applications. Then we presented our crowdsensing based real-time public transport information service, implemented on top of our framework, and its front-end Android application, called TrafficInfo, in detail together with our stop event detector. This detector was developed to automatically detect halt events of public transport vehicles at the stops.

As future work, we plan to develop TrafficInfo further and enhance the different services of all the introduced service levels. Moreover, we intend to recruit a noticeable user base and carry out field experiments with these real users. Their feedback is important to plan the directions for improvements.

ACKNOWLEDGMENT

The publication was supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. Károly Farkas has been partially supported by the Hungarian Academy of Sciences through the Bolyai János Research Fellowship.

REFERENCES

[1] R. Ganti, F. Ye, and H. Lei, “Mobile Crowdsensing: Current State and Future Challenges,” *IEEE Communications Magazine*, pp. 32–39, Nov. 2011.

[2] P. Saint-Andre, “Extensible Messaging and Presence Protocol (XMPP): Core,” RFC 6120 (Proposed Standard), Internet Engineering Task Force, Mar. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6120.txt

[3] R. L. Szabo and K. Farkas, “A Publish-Subscribe Scheme Based Open Architecture for Crowd-sourcing,” in *Proceedings of 19th EUNICE Workshop on Advances in Communication Networking (EUNICE 2013)*. Springer, Aug. 2013, pp. 1–5.

[4] Google Inc., “General Transit Feed Specification Reference.” [Online]. Available: https://developers.google.com/transit/gtfs/reference/

[5] P. Zhou, Y. Zheng, and M. Li, “How Long to Wait?: Predicting Bus Arrival Time with Mobile Phone based Participatory Sensing,” in *Proceedings of the Tenth International Conference on Mobile Systems, Applications, and Services (MobiSys 2012)*, Jun. 2012.

[6] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson, “Cooperative Transit Tracking Using Smart-phones,” in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys 2010)*, Nov. 2010, pp. 85–98.

[3] http://developer.android.com/guide/topics/sensors/index.html

[4] https://developer.android.com/reference/android/location/GpsStatus.NmeaListener.html

[5] http://www.cs.waikato.ac.nz/ml/weka/

[7] L. Bedogni, M. Di Felice, and L. Bononi, “By Train or by Car? Detecting the User’s Motion Type Through Smartphone Sensors Data,” in *Proceedings of IFIP Wireless Days Conference (WD 2012)*, 2012, pp. 1–6.

[8] J. Biagioni, T. Gerlich, T. Merrifield, and J. Eriksson, “EasyTracker: Automatic Transit Tracking, Mapping, and Arrival Time Prediction Using Smartphones,” in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems (SenSys 2011)*, Nov. 2011, pp. 1–14.

[9] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, “Extensible Markup Language (XML) 1.0 (Fifth Edition),” W3C, W3C Recommendation REC-xml-20081126, Nov. 2008. [Online]. Available: http://www.w3.org/TR/2008/REC-xml-20081126/

[10] P. Saint-Andre, “XEP-0045: Multi-User Chat,” XMPP Standards Foundation, Standards Track XEP-0045, Feb. 2012. [Online]. Available: http://xmpp.org/extensions/xep-0045.html

[11] P. Millard, P. Saint-Andre, and R. Meijer, “XEP-0060: Publish-Subscribe,” XMPP Standards Foundation, Draft Standard XEP-0060, Jul. 2010. [Online]. Available: http://xmpp.org/extensions/xep-0060.html

[12] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, “The Many Faces of Publish/Subscribe,” *ACM Comput. Surv.*, vol. 35, no. 2, pp. 114–131, Jun. 2003.

[13] M. M. Haklay and P. Weber, “OpenStreetMap: User-Generated Street Maps,” *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 12–18, Oct. 2008. [Online]. Available: http://dx.doi.org/10.1109/MPRV.2008.80

[14] J. Fogarty, R. S. Baker, and S. E. Hudson, “Case Studies in the Use of ROC Curve Analysis for Sensor-based Estimates in Human Computer Interaction,” in *Proceedings of Graphics Interface 2005*, ser. GI ’05. School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada: Canadian Human-Computer Communications Society, 2005, pp. 129–136. [Online]. Available: http://portal.acm.org/citation.cfm?id=1089508.1089530

Károly Farkas received his Ph.D. degree in Computer Science in 2007 from ETH Zurich, Switzerland, and his M.Sc. degree in Computer Science in 1998 from the Budapest University of Technology and Economics (BME), Hungary. Currently he is working as an associate professor at BME. His research interests cover the field of communication networks, especially autonomic, self-organized, wireless and mobile ad hoc networks, and mobile crowdsourcing. He has published more than 70 scientific papers in different journals, conferences and workshops and he has given a plenty of regular and invited talks. In the years past, he supervised a number of student theses and coordinated or participated in several national and international research projects, such as CityCrowdSource of EIT ICTLabs. Moreover, he acted as program committee member, reviewer and organizer of numerous scientific conferences, thus he took the general co-chair role of the IEEE PerCom 2014 conference and the TPC co-chair role of the CROWDSENSING 2014 and the CASPer 2015 workshops. He is the coordinator of the local Cisco Networking Academy and was the founding initiator of the Cisco IPv6 Training Laboratory and the BME NetSkills Challenge student competition at BME. Between 2012 - 2015 Dr. Farkas has been awarded the Bolyai János Research Fellowship of the Hungarian Academy of Sciences.

Gábor Fehér graduated in 1998 at the Budapest University of Technology and Economics on the Faculty of Electronic Engineering and Informatics. In 2004 he received a PhD. degree, the topic of his thesis was resource control in IP networks. Currently he is an associate professor at the same university. Besides giving lectures, he is also contributing to various national and international research projects. From 2004 he is continuously involved in three consecutive EU founded IST/ICT projects. He has teaching activity on the faculty’s Smart City specialization working with microelectronics, sensor networks and smartphones. He and his students are working on more projects with crowdsouring and crowdsensing, from the basic research up to the prototype applications.

András Benczúr received his Ph.D. at the Massachusetts Institute of Technology in 1997. Since then his interest turned to Data Science. He is the head of 30 doctoral students, post-docs and developers at the Institute for Computer Science and Control of the Hungarian Academy of Sciences (SZTAKI). He is site coordinator in the Hungarian Future-ICT project, and cloud computing activity leader in the Budapest node of EIT ICTLabs. He serves on the program committees of leading conferences including WWW, WSDM, ECML/PKDD, he was Workshop Chair for WWW 2009 and main organizer of the ECML/PKDD Discovery Challenge 2010. In 2012 he was awarded the Momentum grant of the Hungarian Academy of Sciences for his research in Big Data.

Csaba Sidló started working on data warehousing projects and application driven research problems of extremely large data sets in 2000. He joined the Institute for Computer Science and Control of the Hungarian Academy of Sciences (SZTAKI) in 2004; he is now head of the Big Data Business Intelligence research group. His main interest is Big Data analytics and business intelligence on scalable distributed architectures. His industrial projects include master data entity resolution, integration and analytics of log, web and location data. He is currently involved in several big data projects for web, telecom and sensor data analytics. Csaba authored several research papers and book chapters, and has a PhD in Informatics from Eötvös University, Hungary.

Membrane Systems from the Viewpoint of the Chemical Computing Paradigm

Péter Battyányi and György Vaszil

Abstract—**Membrane systems are nature motivated abstract computational models inspired by basic features of biological cells and their membranes. They are examples of the chemical computational paradigm which describes computation in terms of chemical solutions where molecules interact according to rules defining their reaction capabilities. In this survey, we first review some of the basic features and properties of the chemical paradigm of computation, and also give a short introduction to membrane systems. Then examine the relationship of the certain chemical programming formalisms and some simple types of membrane systems.**

Index Terms—**Abstract computational models, chemical computing paradigm, membrane systems.**

I. Introduction

MEMBRANE systems are abstract computational models inspired by the architecture and the functioning of biological cells. Their structure consists of hierarchically embedded membranes, with multisets of symbolic objects associated to the regions enclosed by them. The evolution of the system is governed by rules assigned to the regions. The system performs nondeterministic transformations of these multisets, which produces a series of configuration changes which is interpreted as a computation. The area was initiated by Gh. Păun in [11] and the literature on the domain has grown very fast. Soon it became one of the most important and most popular areas of Natural Computing. For details on the developments, consult the monograph [12] or the more recent handbook [13].

In this survey, we look at the field of membrane computing as a particular example of the so called chemical computational paradigm. This paradigm aims to describe computations in terms of a symbolic chemical solution of molecules and the reactions which can take place between them. Its origins go back to the Gamma programming language of Bânatre and Le Métayer introduced in [6], [7]. Their aim was to free the expression of algorithms from the sequentiality which is not inherently present in the problem to be solved, that is, the sequentiality which is implied by the structure of the computational model on which the given algorithm is to be performed. In other words, their aim was to free the programmer from the necessity of taking into account the underlying architecture of the machine that is being programmed.

The idea was carried on into several directions, see [3] for an overview. From our point of view, one of the most interesting developments was the introduction of the so called chemical abstract machine, see [9], where the notion of membrane appears serving as a delimiter between different types of sub-solutions, forcing the reactions of the sub-solutions to occur in a locally isolated way. This model and the idea of locally delimited regions and membranes was one of the explicit motivations behind membrane systems, as they appear in [11].

In the following we give a short introduction to some of the formalisms used to describe computations in the chemical way, and also present some of the basic notions of membrane computing. Then, based on the results of [10] and [8] we present some ideas on how the chemical formalisms and membrane systems can be related to each other. This approach is interesting in at least two ways. By being able to translate chemical programs to membrane systems, we could obtain a high level programming language for the description of membrane algorithms. On the other hand, by being able to describe membrane computations with some of the chemical formalisms, we would be able to reason about the properties of membrane systems in a mathematically precise manner.

II. Preliminary Definitions and Notation

An alphabet is a finite non-empty set of symbols V, the set of strings over V is denoted by V^* .

A finite multiset over an alphabet V is a mapping $M : V \to \mathbb{N}$ where $\mathbb{N}$ denotes the set of non-negative integers, and $M(a)$ for $a \in V$ is said to be the multiplicity of a in M. The set of all finite multisets over the set V is denoted by $\mathcal{M}(V)$.

We usually enumerate the not necessarily distinct elements $a_1, \dots, a_n$ of a multiset as $M = \langle a_1, \dots, a_n \rangle$, but the multiset M can also be represented by any permutation of a string $w = a_1^{M(a_1)} a_2^{M(a_2)} \dots a_n^{M(a_n)} \in V^*$, where if $M(x) \neq 0$, then there exists j, $1 \leq j \leq n$, such that $x = a_j$. The empty multiset is denoted by $\emptyset$.

For more on the basics of formal language theory and Membrane Computing the reader is referred to the monograph [15], and the handbooks [14] and [13].

III. Computation as Reactions in a Chemical Solution

A chemical “machine” can be thought of as a symbolic chemical solution where data can be seen as molecules and operations as chemical reactions. If some molecules satisfy a reaction condition, they are replaced by the result of the reaction. If no reaction is possible, the program terminates. Chemical solutions are represented by multisets. Molecules interact freely according to reaction rules which results in an

Manuscript received October 23, 2014, revised December 8, 2014.

Péter Battyányi and György Vaszil are with the Department of Computer Science, Faculty of Informatics, University of Debrecen, Kassai út 26, 4028 Debrecen, Hungary, e-mail: battyanyi.peter@inf.unideb.hu, vaszil.gyorgy@inf.unideb.hu

TABLE I
THE ELEMENTS OF THE ABSTRACT MACHINE AND THE INGREDIENTS OF A CHEMICAL SOLUTION.

Abstract machine	Chemistry
Data	Molecule
Multiset	Solution
Parallelism/nondeterminism	Brownian motion
Computation	Reaction

implicitly parallel, non-deterministic, distributed model. The chemical analogy is carried over also to the execution model: The Brownian motion of the chemical molecules correspond to the parallel and nondeterministic computation of the chemical machine. See table I for a summary of this correspondence.

To help the easier understanding of the notions, we start with the discussion of the Higher-order Chemical Language (HOCL) from [1], which can be presented in a more reader-friendly manner as the mathematically more precise γ-calculus, see [4], which we will also discuss later.

In general, a reaction rule can be written as

$$\textbf{replace}\ P\ \textbf{by}\ M\ \textbf{if}\ C$$

where *P* is a pattern, *C* is the reaction condition, and *M* is the result of the reaction. For example, the solution

$$\langle(\textbf{replace}\ x, y\ \textbf{by}\ x\ \textbf{if}\ x < y),\ 2,\ 7,\ 4,\ 3,\ 6,\ 8\rangle$$

will result in the solution

$$\langle(\textbf{replace}\ x, y\ \textbf{by}\ x\ \textbf{if}\ x < y),\ 2\rangle$$

containing the reaction and the minimum value among the molecules. Notice that the order in which the reactions are performed, that is, the order in which the numbers are compared is not specified.

Solutions can also contain sub-solutions, as seen in the following example, where the least common multiple of 4 and 6 is computed.

Example 1: Let us start with

$$\begin{aligned}&\textbf{let}\ \text{multiplier} = \textbf{replace}\ x, \omega\ \textbf{by}\ \omega\ \textbf{if}\\ &\qquad\qquad\qquad\qquad \text{not}(4\ \text{div}\ x\ \text{and}\ 6\ \text{div}\ x),\\ &\textbf{let}\ \text{clean} = \textbf{replace-one}\ \langle\text{multiplier}, \omega\rangle\ \textbf{by}\ \omega,\\ &\textbf{let}\ \text{min} = \textbf{replace}\ x,\ y\ \textbf{by}\ x\ \textbf{if}\ x < y,\end{aligned}$$

and consider the following solution

$$\begin{aligned}&\langle \text{min, clean, } \langle \text{multiplier, } 10,\ 11,\ 12,\ 13,\ 14,\ 15,\\ &\qquad 16,\ 17,\ 18,\ 19,\ 20,\ 21,\ 22,\ 23,\ 24\rangle\ \rangle\end{aligned}$$

where the "top level" is a solution containing the reactions *min, clean* and a sub-solution, which is another solution with the reaction *multiplier* and a set of numbers. When the sub-solution becomes inert, that is, when the common multiples of 4 and 6 are selected, the reactions of the top level, *min* or *clean* are activated. First only the condition of *clean* can be matched, so it is applied, and the remaining numbers from the sub-solution are "moved" one level higher while the sub-solution and the reaction *multiplier* are eliminated. Now the conditions of *min* can also be matched, resulting in the solution

$$\langle \text{min, } 12\rangle.$$

Notice the pattern ω which is special in the sense that it can match anything, and the "one-shot" reaction *clean* using the keyword **replace-one**, meaning that its application is only possible once, as the application "consumes" the reaction itself.

Based on these examples, the reader might agree that we can formulate some of the important characteristic properties of the chemical computational model as follows:

- *Parallel execution*: when two reactions involve distinct elements, they can occur simultaneously,
- *mutual exclusion*: a molecule cannot take part in more than one reaction at the same time,
- *atomic capture*: either all ingredients of the reaction are present, or no reaction occurs.

Example 2: To see why these characteristics are important, consider the problem of the dining philosophers, a common example for the demonstration of concurrent algorithm design techniques. Let us state the problem as follows: There are 5 philosophers sitting at a round table, with 5 plates of spaghetti in front of them, and 5 forks between the plates. A philosopher is either thinking or eating, but when he is eating, he needs both of the forks on each side of his plate since philosophers only eat spaghetti with two forks. Thus, it is not possible that two neighbors eat at the same time.

A description of the problem can be given in the above described chemical formalism as follows. Let

$$\begin{aligned}&\textbf{let}\ \text{eat} = \textbf{replace}\ Fork : f_1, Fork : f_2\ \textbf{by}\ Phil : f_1\ \textbf{if}\\ &\qquad\qquad\qquad\qquad\qquad f_2 = f_1 + 1\ \text{mod}\ 5,\\ &\textbf{let}\ \text{think} = \textbf{replace}\ Phi : f\ \textbf{by}\\ &\qquad\qquad Fork : f, Fork : f + 1\ \text{mod}\ 5\ \ \textbf{if}\ true,\end{aligned}$$

and consider the following symbolic solution

$$\langle \text{eat, think, } Fork : 1,\ Fork : 2, \ldots, Fork : 5\rangle$$

which contains two reaction rules and five numbered objects of the type *Fork*, representing the situation when all the forks are on the table, that is, when no philosopher is eating. This situation can change through the reaction *eat* which replaces to adjacent forks with the corresponding numbered object of the type *Phil*. Conversely, the reaction *think* replaces an eating philosopher with the corresponding forks.

Consider now the consequences of the above mentioned three characteristic properties for the behavior of this setup. Due to *parallel execution*, if two philosophers are not neighbors at the table, they are allowed to eat simultaneously and independently of each other. The *mutual exclusion* property guarantees that one fork is used by at most one philosopher, and as the consequence of *atomic capture*, deadlocks are "automatically" avoided, since a fork can be picked up by a philosopher only in the case when the other fork is also available.

Now, before we continue, based mainly on [5], we present a more rigorous formalism which we will extend in Section V. As we have already seen, it is a tool for multiset manipulation, and the programs are collections of pairs of reaction conditions and actions. The Γ function is defined as

$$\Gamma((R_1, A_1), \ldots, (R_k, A_k))(M) =$$

$$= \begin{cases} \Gamma((R_1,A_1),\ldots,(R_k,A_k))((M\backslash(x_1,\ldots,x_n)) \\ \qquad\qquad\qquad\qquad \cup A(x_1,\ldots,x_n)), \\ \qquad \text{if } x_1,\ldots,x_n \in M \text{ and } R_i(x_1,\ldots,x_n) \text{ for} \\ \qquad \text{some } 1 \le i \le k, \text{ or} \\ M, \quad \text{otherwise,} \end{cases}$$

where R_i and A_i, $1 \le i \le k$, are n-ary relations (the reaction conditions) and n-ary functions (the actions) on the elements of the multiset M, respectively. If some elements of the multiset M, say $x_1,\ldots,x_n$, satisfy a reaction condition R_i for some i, $1 \le i \le k$, then the elements may be replaced by the result of the action $A_i(x_1,\ldots,x_n)$, and the Γ function is applied on the resulting multiset again. This process continues until no elements satisfy any of the relations R_i, $1 \le i \le k$.

Example 3: To clarify this type of notation, consider the Gamma program for selecting the minimal element of a set of numbers.

$$\begin{aligned} minset(M) &= \Gamma(R,A)(M) \text{ where} \\ & \quad R(x,y) = (x < y), \\ & \quad A(x,y) = (x). \end{aligned}$$

IV. MEMBRANE SYSTEMS

Similarly to chemical programs, membrane systems (also called P systems) work with multisets of symbolic objects. They consist of a structured set of regions, each containing a multiset of objects and a set of evolution rules which define how the objects are produced, destroyed, or moved inside the system. A computation is performed by passing from one configuration to another one, applying the rules synchronously in each region.

We consider two variants in this paper: the rules are either multiset rewriting rules given in the form of $u \to v$, or communication rules of the form $(u, in; v, out)$ where u, v are finite multisets. In both cases, the rules are applied in the maximal parallel manner, that is, as many rules are applied in each region as possible. The end of the computation is defined by halting: the computation finishes when no more rules can be applied in any of the regions. The result is a number, the number of objects in a membrane labeled as output.

A *P system* of degree $n \ge 1$ is a construct

$$\Pi = (O, \mu, w_1, \ldots, w_n, R_1, \ldots, R_n, out)$$

where

- O is an alphabet of objects,
- μ is a membrane structure of the n membranes. The outmost membrane which is unique and usually labeled with 1, is called the skin membrane, and the membrane structure is denoted by a sequence of matching parentheses where the matching pairs have the same label as the membranes they represent, see Figure 1 for an example.
- $w_i \in \mathcal{M}(O)$, $1 \le i \le n$, are the initial contents of the n regions.
- R_i, $1 \le i \le n$, are the sets of evolution or communication rules associated to the regions, and
- $out \in \{1, \ldots, n\}$ is the label of the output membrane.

As mentioned above, we consider two types of P systems in this paper. In the case of *rewriting P systems*, the rules of the set R are of the form $u \to v$ where $u \in \mathcal{M}(O)$ and $v \in \mathcal{M}(O \times TAR)$ with $TAR = \{here, out\} \cup \{in_j \mid 1 \le j \le n\}$. In the case of *antiport P systems*, the rules don't allow the changing of the objects only their movement between the regions, they are of the form $(u, in; v, out)$ where $u, v \in \mathcal{M}(O)$.

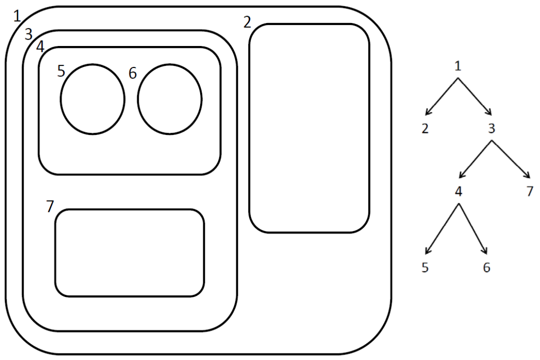


Fig. 1. A membrane structure with the skin membrane labeled as 1, and the corresponding tree representation. It can also be represented by a string of matching parentheses as $[_1\ [_2\ ]\ [_3\ [_4\ [_5\ ]\ [_6\ ]\ ]\ [_7\ ]\ ]\ ]$.

The rules are applied in the non-deterministic, maximally parallel manner to the n-tuple of multisets of objects constituting the configuration of the system. For two configurations $C_1 = (u_1, \ldots, u_n)$ and $C_2 = (v_1, \ldots, v_n)$, we can obtain C_2 from C_1, denoted as $C_1 \Rightarrow C_2$, by applying the rules of $R_1, \ldots, R_n$ in the following way.

In the case of rewriting P systems, the application of $u \to v \in R_i$ in the region i means to remove the objects of u from u_i and add the new objects specified by v to the system. The objects of v should be added to the regions as specified by the target indicators associated to them: If v contains a pair $(a, here) \in O \times TAR$, then a is placed in region i, the region where the rule is applied. If v contains $(a, out) \in O \times TAR$, then a is added to the contents of the parent region of region i; if v contains $(a, in_j) \in O \times TAR$ for some region j which is contained inside the region i (so region i is the parent region of region j), then a is added to the contents of region j.

In the case of antiport systems, the application of $(u, in; v, out) \in R_i$ in region i means to move the objects of u from the parent region into region i, and simultaneously, to move the objects of v into the parent region.

The n-tuple $(w_1, \ldots, w_n)$ is the initial configuration of Π.

The objects evolve simultaneously, and the rules by which they evolve are chosen nondeterministically, but in a maximally parallel manner. This means, that in each region, objects are assigned to rules, nondeterministically choosing the rules and the objects assigned to each rule, but in such a way that no further rule can be applied to the remaining objects. A rule can be applied in the same step more than once, only the number of occurrences of objects matters. Objects which remain unassigned, appear unchanged in the next configuration.

A sequence of transitions between configurations is called a computation. A computation is successful if it halts, that is, if it reaches a configuration where no application of any of the rules are possible. In this case, the result is the multiset of objects which is present in the output region in the halting configuration.

As an example, let us consider how the dining philosopher problem can be represented in the P system framework.

Example 4: Consider the antiport P system

$$\Pi = (O, \mu, w_0, w_1, \dots, w_5, R_0, R_1, \dots, R_5, out)$$

where $O = \{t_i, e_i, f_i \mid 1 \leq i \leq 5\}$, $w_0 = e_1 f_1 \dots e_5 f_5$, $w_i = t_i,\ 1 \leq i \leq 5$. The sets of rules are defined as $R_0 = \emptyset$, and for $1 \leq i \leq 5$ as

$$\begin{aligned} R_i \quad = \quad & \{(e_i f_i f_{i+1\ mod\ 5}, in; t_i, out),\ (e_i, in; e_i, out), \\ & (t_i, in; e_i f_i f_{i+1\ mod\ 5}, out),\ (t_i, in; t_i, out)\}. \end{aligned}$$

There are five regions, labeled by 1 to 5 in this system enclosed in a sixth one, the skin membrane, which is labeled by 0. The initial configuration, when the enclosed regions $i,\ 1 \leq i \leq 5$ contain the objects $t_i,\ 1 \leq i \leq 5$, respectively, corresponds to the situation when all philosophers are thinking. Applying the rule $(t_i, in; t_i, out)$, the ith philosopher may keep thinking, or applying the rule $(e_i f_i f_{i+1\ mod\ 5}, in; t_i, out)$, he may start eating.

As the properties of *parallel execution*, *mutual exclusion*, and *atomic capture* also hold in the case of membrane systems, the above given description of the dining philosophers also have the same desirable properties as the HOCL description given in the previous section.

Let us consider now an example from [12]. To this aim we introduce two features that we did not consider so far: *priorities* among the rules, and membrane *dissolution*.

Example 5: Let Π the following system of three membranes.

$$\Pi = (O, [_1\ [_2\ [_3\ ]\ ]\ ], \emptyset, \emptyset, af, R_1, R_2, R_3, 1)$$

where $O = \{a, b, d, e, f\}$, and the sets of rules are defined as

$$\begin{aligned} R_1 &= \{d \to \emptyset\}, \\ R_2 &= \{b \to d, d \to de\} \cup \{ff \to f > f \to \delta\}, \\ R_3 &= \{a \to ab, a \to b\delta, f \to ff\}. \end{aligned}$$

Priorities are introduced in the rule set R_2, denoted by the relation $ff \to f > f \to \delta$, which means that as long as the rule $ff \to f$ is applicable, $f \to \delta$ cannot be applied. Membrane dissolution is also introduced here by the symbol δ. When the rule $f \to \delta$ is used, the corresponding membrane (the membrane surrounding region 2 in this case) is dissolved/removed, and the objects it contains become elements of the parent region (region 1 in this case).

The computation of Π starts in the initial configuration $(\emptyset, \emptyset, af)$. Applying the rules $a \to ab$ and $f \to ff$ of R_3, we get $(\emptyset, \emptyset, abff)$ after one computational step. Repeating this for another $k-2$ steps, we get

$$(\emptyset, \emptyset, abff) \Rightarrow \dots \Rightarrow (\emptyset, \emptyset, ab^{k-1} f^{2^{k-1}}).$$

Now, if we apply $a \to b\delta$ instead of $a \to ab$, the membrane delimiting region 3 is dissolved, so we arrive to the configuration $(\emptyset, b^k f^{2^k})$. Next, we can apply the rules of R_2 to the bs and fs, resulting in $(\emptyset, d^k f^{2^{k-1}})$ after the next step, by replacing all bs with ds and halving the number of fs in parallel. Note that as long as there are more than two f symbols, the rule $f \to \delta$ cannot be applied, because $ff \to f$ has higher priority. Applying $d \to de$ and $ff \to f$ as long as possible, we obtain

$$(\emptyset, d^k e^k f^{2^{k-2}}) \Rightarrow (\emptyset, d^k e^{2k} f^{2^{k-3}}) \Rightarrow \dots \Rightarrow (\emptyset, d^k e^{(k-1)k} f)$$

and then $(d^k e^{kk})$ by applying the dissolution rule $f \to \delta$. Now if the rule of R_1 is applied erasing all d symbols, we obtain the configuration (e^{k^2}), and the system halts. The result of this computation is the number k^2. We can observe that the number k^2 can be computed by the system for any k in a similar manner, thus, the set of numbers computed by Π is the set $N(\Pi) = \{k^2 \mid k \geq 1\}$.

From a theoretical point of view membrane systems are both powerful and efficient computational devices. Powerful, as many of their variants are computationally complete, that is, equivalent in computational power to Turing machines, and efficient as, due to their parallelity and "chemical nature", they are able to provide efficient solutions to computationally hard (typically NP complete, or even PSPACE complete) problems. More details and further references can be found in [13].

V. Chemical programs and Membrane Systems

As we have seen in the previous sections, membrane systems and programs written in the Gamma language are closely related. This is not surprising because they both provide a realization of what we call the chemical paradigm of computing. They both work with symbolic chemical solutions which are represented by multisets, containing molecules which interact freely according to given reaction rules, resulting in a parallel, nondeterministic, distributed model. In this section we turn to the demonstration of links between the two formalisms.

A. Describing chemical programs by membranes systems

First we demonstrate how membrane systems could mimic the behavior of systems which are described by chemical programs. To this aim, we review the approach of [10], recalling an example from [1] which gives the chemical description of a mail system.

The mail system (see Figure 2) is described by a solution. Messages exchanged between clients are represented by basic molecules.

- Solutions named $ToSend_{d_i}$ contain the messages to be sent by the client i of domain d.
- Solutions named$Mbox_{d_i}$ contain the messages received by the client i of domain d.
- Solutions named $Pool_d$ contain the messages that the server of domain d must take care of.
- The solution named Network represents the global network interconnecting domains.
- A client i in domain d is represented by two active molecules $send_{d_i}$ and $recv_{d_i}$.
- A server of a domain d is represented by two active molecules put_d and get_d.

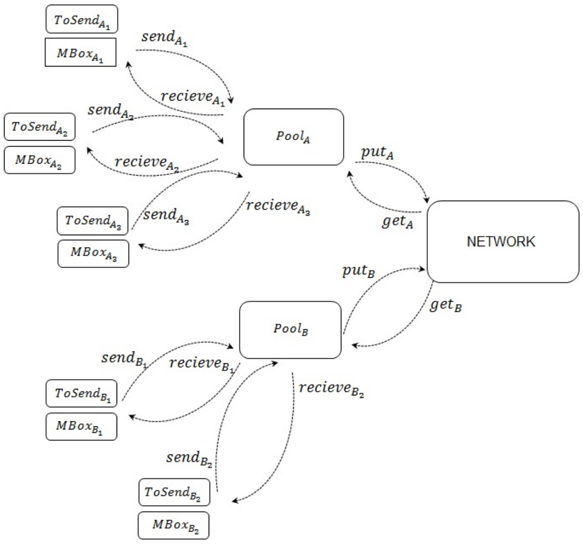


Fig. 2. The mail system described in Section V.

Clients send messages by adding them to the pool of messages of their domain. They receive messages from the pool of their domain and store them in their mailbox. Message stores are represented by sub-solutions.

The movement of messages are performed by reaction rules of the form

$$\textbf{replace } A:\langle msg,\omega_A\rangle,\ B:\langle\omega_B\rangle \quad \textbf{by } A:\langle\omega_A\rangle,\ B:\langle msg,\omega_B\rangle \textbf{ if } Cond.$$

The $send$ molecule sends the messages from the client to the pool, $recv$ gets the messages from the pool and places them inside the message box of the client, put forwards messages to the network, get receives messages from the network.

$$send_{d_i} = \textbf{replace } ToSend_{d_i}:\langle msg,\omega_t\rangle, Pool_d:\langle\omega_p\rangle \quad \textbf{by } ToSend_{d_i}:\langle\omega_t\rangle, Pool_d,:\langle msg,\omega_p\rangle$$

$$recv_{d_i} = \textbf{replace } Pool_d:\langle msg,\omega_p\rangle, MBox_{d_i}:\langle\omega_b\rangle \quad \textbf{by } Pool_d:\langle\omega_p\rangle, MBox_{d_i}:\langle msg,\omega_b\rangle \quad \textbf{if } recipient(msg)=i$$

$$put_d = \textbf{replace } Pool_d:\langle msg,\omega_p,\rangle, Network:\langle\omega_n\rangle \quad \textbf{by } Pool_d:\langle\omega_p\rangle, Network:\langle msg,\omega_n\rangle \quad \textbf{if } recipientDomain(msg)\neq d$$

$$get_d = \textbf{replace } Network:\langle msg,\omega_n\rangle, Pool_d:\langle\omega_p\rangle \quad \textbf{by } Network:\langle\omega_n\rangle, Pool_d:\langle msg,\omega_p\rangle \quad \textbf{if } recipientDomain(msg)=d$$

The solution representing the mail system contains the above described molecules together with sub-solutions representing the messages to be sent and received (called $ToSend$ and $MBox$, respectively) for each user, A_1, A_2, A_3, and B_1, B_2.

MailSystem:

$$\begin{aligned}\langle & send_{A_1}, recv_{A_1}, ToSend_{A_1}:\langle\ldots\rangle, MBox_{A_1}:\langle\ldots\rangle,\\ & send_{A_2}, recv_{A_2}, ToSend_{A_2}:\langle\ldots\rangle, MBox_{A_2}:\langle\ldots\rangle,\\ & send_{A_3}, recv_{A_3}, ToSend_{A_3}:\langle\ldots\rangle, MBox_{A_3}:\langle\ldots\rangle,\\ & put_A, get_A, Pool_A, Network, put_B, get_B, Pool_B,\\ & send_{B_1}, recv_{B_1}, ToSend_{B_1}:\langle\ldots\rangle, MBox_{B_1}:\langle\ldots\rangle,\\ & send_{B_2}, recv_{B_2}, ToSend_{B_2}:\langle\ldots\rangle, MBox_{B_2}:\langle\ldots\rangle\ \rangle\end{aligned}$$

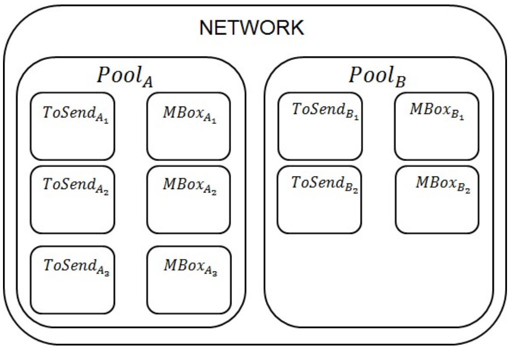


Fig. 3. The membrane mail system corresponding to the system of Figure 2

This chemical solution can be represented by a membrane system where message stores are represented by membranes, active molecules (or reactions) are represented by evolution rules. If we denote the regions as in Figure 3, and messages addressed to recipient d_i are represented by objects msg_{d_i}, we need the following rules:

- $R_{ToSend_{d_i}} = \{msg_{d_j} \to (msg_{d_j}, out)\}$,
- $R_{Pool_{d_i}} = \{msg_{d'_j} \to (msg_{d'_j}, out),$
  $msg_{d_i} \to (msg_{d_i}, in_{MBox_i})\}$, and
- $R_{Network} = \{msg_{d_i} \to (msg_{d_i}, in_{Pool_d})\}$.

The rules corresponding to the "outbox" of users send the messages to their pool, and the rules corresponding to the pools, $Pool_A$ and $Pool_B$, place them into the $MBox$ of the user. If a message is addressed to a user belonging to the other pool, then it is sent to the network which forwards it to the corresponding message pool.

B. Describing membrane systems by chemical programs

Let us know continue with considerations in the "opposite direction", namely, with the study of how membrane computations could be described with a chemical program. To this aim, we summarize the results contained in [8].

First we introduce some elements of the γ-calculus of Banâtre and his coauthors, see for example [4]. Similarly to the chemical programming language used in the previous section, it is a higher order extension of the Gamma formalism. We need it in order to be able to have a calculus, a mathematically precise description of chemical computations.

The main rule of the calculus is the reaction rule

$$\gamma(P)[C].M, N \to \phi M$$

where P is a pattern, C is a condition, M is the result, and N is the multiset to which the rule is applied. Its application produces ϕM, where $match(P,N)=\phi$ and ϕ assigns values to variables in such a way that $\phi(C)$ is true.

Without further clarifications, let us look at the following example. The γ-term

$$\gamma(x,y)[x\le y].y,(3,4)$$

reduces to the multiset (4) since in order for $\phi(x \leq y)$ to hold, $match((x,y),(3,4)) = \phi$ should be such, that $\phi = \{x \mapsto 3, y \mapsto 4\}$. This means that $\phi(y) = 4$, thus, the result of the reaction $\gamma(x,y)[x \leq y].y,(3,4)$ is the multiset (4).

There calculus has several other reduction rules which we do not discuss here, but we do recall that a $replace$ operator can also be defined, which behaves similarly to the instruction with the same name used in the chemical programming language we saw in the previous sections. This is used in the following example which calculates the largest prime which is less than or equal to 6.

Example 6:

$largestprime(6) =$
let $sieve = replace$ $(\langle x\rangle, \langle y\rangle)$ by $\langle x\rangle$ if x div y in
let $max = replace$ $(\langle x\rangle, \langle y\rangle)$ by $\langle x\rangle$ if $x \leq y$ in
$\langle\ \langle\langle 2\rangle, \langle 3\rangle, \ldots, \langle 6\rangle, sieve\rangle, \gamma(\langle x\rangle)[true](x, max)\ \rangle$

The pattern standing in the last term $\gamma(\langle x\rangle)[true](x, max)$ is a solution $\langle x\rangle$, which can be only matched by inert solutions, thus, first the prime numbers are selected with the term $sieve$, producing the inert solution $\langle\langle 2\rangle, \langle 3\rangle, \langle 5\rangle, sieve\rangle$ which matches the pattern $\langle x\rangle$ in the γ-term, resulting in $\langle\langle 2\rangle, \langle 3\rangle, \langle 5\rangle, sieve, max\rangle$, and now max chooses the maximum among them, ending up with $\langle\langle 5\rangle, sieve, max\rangle$.

Using these ingredients, we can define a γ-term which is able to "simulate" the computations of a membrane system. More precisely, for each configuration C of a membrane system Π, we can define a term which contains the objects of the configuration together with *replace* operators corresponding to the rules of Π (and several technical details which we don't discuss) which enables the reduction in the calculus to proceed in such a way that it reproduces the results of the maximally parallel rule application of the P system. Namely, we have the following theorem, see [8].

Theorem 1: Let Π be a membrane system, and $C_0, C_1, \ldots, C_m$ be a sequence of configurations denoting a terminating computation.

Then there exists a γ-term $M(\Pi)$, and a reduction sequence $M(\Pi) \rightarrow M_1 \rightarrow \ldots \rightarrow M_s$, such that M_s cannot be further reduced, and if $C_m = (w_1, \ldots, w_n)$, then for all objects $a \in O$ and regions i, $1 \leq i \leq n$, M_s contains the same number of copies of (a, i), as w_i contains a.
In effect, the theorem establishes for the sequence $\Pi_0, \Pi_1, \ldots, \Pi_m$ of P-systems corresponding to the computation $C_0, C_1, \ldots, C_m$ starting from $\Pi = \Pi_0$ a sequence $M(\Pi) \rightarrow M_1 \rightarrow \ldots \rightarrow M_s$ of γ-terms such that there is an index set $0 < k_1 < \ldots < k_n = s$ with the property $M(\Pi_j) = M_{k_j}$ for $1 \leq j \leq n$.

VI. Conclusion

First we have briefly reviewed the chemical computing paradigm and the notion of membrane systems, then we have discussed their relationship by describing results from [10] and [8]. These results represent the first steps in the direction of establishing the relationship of the two paradigms. This approach could be interesting from several points of view. By being able to translate chemical programs to membrane systems, we could obtain a high level programming language for the description of membrane algorithms. By being able to describe membrane computations with a mathematically precise chemical calculus, we could use it to reason about the properties of membrane systems in a mathematical way.

Acknowledgment

The authors acknowledge the support of the European Union through the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project which is co-financed by the European Social Fund.

References

[1] J.P. Banâtre, P. Fradet, and Y. Radenac, Higher-order chemical programming style. In: [2], 84–95.

[2] J.P. Banâtre, P. Fradet, J.L. Giavitto, and O. Michel, editors, *Unconventional Programming Paradigms, International Workshop UPP 2004, Le Mont Saint Michel, France, September 15-17, 2004, Revised Selected and Invited Papers*. Volume 3566 of *Lecture Notes in Computer Science*, Springer, Berlin Heidelberg, 2005.

[3] J.-P. Banâtre, P. Fradet, D. Le Métayer: Gamma and the chemical reaction model: Fifteen years after. In *Multiset Processing. Mathematical, Computer Science, and Molecular Computing Points of View*. Volume 2235 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2001, 17–44.

[4] J.P. Banâtre, P. Fradet, Y. Radenac, Principles of chemical computing. *Electronic Notes in Theoretical Computer Science* 124 (2005) 133–147.

[5] J.P. Banâtre, P. Fradet, Y. Radenac, Generalized multisets for chemical programming. *Mathematical Structures in Computer Science* 16(4) (2006), 557 – 580.

[6] J.P. Banâtre, D. Le Métayer, A new computational model and its discipline of programming. *Technical Report RR0566*, INRIA (1986).

[7] J.P. Banâtre, D. Le Métayer, Programming by multiset transformation. *Communications of the ACM* 36 (1993), 98–111.

[8] P. Battyányi, Gy. Vaszil, Describing membrane computations with a chemical calculus. *Fundamenta Informaticae* 134 (2014), 39–50.

[9] G. Berry, G. Boudol, The chemical abstract machine. *Theoretical Computer Science* 96 (1992), 217–248.

[10] M. Fésüs, Gy. Vaszil, Chemical programming and membrane systems. In: *Proc. 14th International Conference on Membrane Computing*, Institute of Mathematics and Computer Science, Academy of Moldova, Chişinău, 2013, 313–316.

[11] Gh. Păun, Computing with membranes. *Journal of Computer and System Sciences* 61 (2000), 108–143.

[12] Gh. Păun, Membrane Computing - An Introduction. Springer-Verlag, Berlin 2002.

[13] Gh. Păun, G. Rozenberg, A. Salomaa (eds): The Oxford Handbook of Membrane Computing, Oxford University Press (2010)

[14] G. Rozenberg, A. Salomaa (eds): Handbook of Formal Languages, Springer Berlin (1997)

[15] A. Salomaa: Formal Languages, Academic Press, New York (1973)

Péter Battyányi received his MSc in mathematics in 1997 at the University of Debrecen. He conducted his PhD studies in 2005-2007 at the Université de Savoie, France. Since 2008, he is an assistant professor at University of Debrecen. His main fields of interest are mathematical logic, logical calculi, computability, artificial intelligence, formal verification of programs, models of parallel programming.

György Vaszil is head of the Department of Computer Science at the Faculty of Informatics of the University of Debrecen. He received his PhD at the Eötvös Loránd University, Budapest, in 2001, and the title Doctor of the Hungarian Academy of Sciences in 2014. His research interests include formal languages and automata theory, unconventional and nature motivated computational models and architectures.

Factorization models for context-aware recommendations

Balázs Hidasi

Abstract—**The field of implicit feedback based recommender algorithms have gained increased interest in the last few years, driven by the need of many practical applications where no explicit feedback is available. The main difficulty of this recommendation task is the lack of information on the negative preferences of the users that may lead to inaccurate recommendations and scalability issues. In this paper, we adopt the use of context-awareness to improve the accuracy of implicit models—a model extension technique that was applied successfully for explicit algorithms. We present a modified version of the iTALS algorithm (coined iTALSx) that uses a different underlying factorization model. We explore the key differences between these approaches and conduct experiments on five data sets to experimentally determine the advantages of the underlying models. We show that iTALSx outperforms the other method on sparser data sets and is able to model complex user–item relations with fewer factors.**

Index Terms—**context-awareness, implicit feedback, model comparison, recommender systems, tensor factorization.**

I. INTRODUCTION

Recommender systems are information filtering tools that help users in information overload to find interesting items. For modeling user preferences, classical approaches either use item metadata (content based filtering, CBF; [1]), or user–item interactions (collaborative filtering, CF; [2]). CF algorithms proved to be more accurate than CBF methods, if sufficient interaction data (or *events*) is available [3]. CF algorithms can be further divided into memory and model based algorithms. An important subclass of the latter is the factorization algorithms (e.g. matrix factorization).

Latent factor based CF methods gained popularity due to their attractive accuracy and scalability [4]. They intend to capture user preferences by uncovering latent features that explain the observed user–item events (ratings). Models are created by the factorization of the partially observed user–item rating matrix, and the user preferences are approximated by the scalar product of the user and item factors. Matrix factorization (MF) methods may differ in the learning method and the objective function. For learning, MF methods may apply, e.g., alternating least squares (ALS; [5]), stochastic gradient [6], or a probabilistic framework [7].

Depending on the nature of the user–item interactions, recommendation problems can be classified into explicit and implicit feedback based problems. In the former case, users provide explicit information on their preferences, typically in form of ratings. In the latter case, user preferences are captured seamlessly via user activity. Implicit feedback algorithms use user interactions like viewing and purchasing retrieved e.g. from website usage logs. Obviously, implicit feedback data is less reliable because the presence of an action is only an uncertain implication that the user likes the item and the absence of an action rarely means negative preference.

The implicit problem is much more important in practical applications than the explicit one, because most of the users of online e-commerce shops or services do not tend to rate items even if such an option is available[8], because (1) when purchasing they have no information on their satisfaction rate (2) they are not motivated to return later to the system to do so. In such a case, user preferences can only be inferred by interpreting user actions (also called *events*). For instance, a recommender system may consider the navigation to a particular item page as an implicit sign of preference for the item [9]. The user history specific to items are thus considered as implicit feedback on the user's taste. Note that the interpretation of implicit feedback data may not necessarily reflect user preferences which makes the implicit feedback based preference modeling a much harder task. For instance, a purchased item could be disappointing for the user, so it might not mean a positive feedback. The strength of the events' indication of preferences varies on a type by type basis. E.g. purchasing an item is a stronger indicator than looking at a product page (browsing). Missing navigational or purchase information can not be interpreted as negative feedback. The absence of the negative feedback forces us to use the information stored in the "missing" events. Most (explicit) algorithms iterate over the known ratings and use gradient descent to minimize the error function. This is not applicable in the implicit case as the number of known ratings is equal to all possible ratings as we should use the "missing" events as well. Although the explicit feedback problem is much thoroughly studied research topic, in the last few years implicit feedback algorithms have gained increased interest thanks to its practical importance; see [8], [10], [11].

Classical collaborative filtering methods only consider direct user–item interaction data to create the model. However, we may have additional information related to items, users or events, which are together termed *contextual information*. Context can be, for instance, the time or location of recommendation. Any additional information to the user–item interaction can be considered as context. Here we assume that the context dimensions are event contexts, meaning that their value is not determined solely by the user or the item; rather it is bound to the transaction itself. E.g. the time of the event is an event context, while the genres of the item is not. Integrating context into the recommender model improves the model capacity and

Manuscript received August 28, 2014, revised December 10, 2014. The author is affiliated with Gravity Research and Development Inc. and with the Budapest University of Technology and Economics. Email: balazs.hidasi@gravityrd.com.

increases accuracy, and became therefore a popular approach for the explicit algorithms recently. We argue that implicit algorithms can benefit *even more* from the context due to the uncertainty of the user feedback.

Context-aware recommendation algorithms can be divided into three groups [12]: (1) pre-filtering approaches partition the training data according to the value of the context(s) and train traditional (non context-aware) algorithms on said partitions; (2) post-filtering approaches disregard the context during training, but modify the list of recommendations according to the actual context-state; (3) contextual modeling approaches consider the context dimension(s) during the learning process.

In this paper, we use the contextual modeling approach. More specifically we extend factorization methods with context dimensions. To incorporate the context in factorization methods, the underlying model needs to be modified. However the model can be modified in several ways and each of these implies a conceptually different view on the role of the context. Building on our previous work, we present a variant of the (context-aware) iTALS algorithm [13] – coined iTALSx[1] – that uses a different underlying model.

The rest of the paper is organized as follows. Section II gives a brief overview on context-aware recommender systems. The iTALSx method is presented in Section III. The key conceptual differences between iTALS and iTALSx are highlighted in Section IV. The experimental comparison of the two approaches (conducted on five implicit feedback data sets) are described in Section V. Finally, Section VI concludes this work.

A. Notation

We will use the following notation in the rest of this paper:

- $A \circ B \circ \ldots$: The Hadamard (elementwise) product of A, B, $\ldots$. The operands are of equal size, and the result's size is also the same. The element of the result at index $(i, j, k, \ldots)$ is the product of the element of A, B, $\ldots$ at index $(i, j, k, \ldots)$.
- A_i: The i^{th} column of matrix A.
- $A_{i_1,i_2,\ldots}$: The $(i_1, i_2, \ldots)$ element of tensor/matrix A.
- K: The number of features, the main parameter of the factorization.
- D: The number of dimensions of the tensor.
- T: A D dimensional tensor that contains only zeroes and ones (preference tensor).
- W: A tensor with the same size as T (weight/confidence tensor).
- $S_{<X>}$: The size of T in dimension X (e.g. $< X >= U$ (Users)).
- N^+: The number of ratings (explicit case); non-zero elements in tensor T (implicit case).
- U, I, C: A $K \times S_{<X>}$ sized matrices. Its columns are the feature vectors for the entities in the user/item/context dimension.
- R: Training data that contains (u, i, c) triplets i.e. user–item–contex-state combinations.

II. Context-aware recommender systems

Context-aware recommender systems (CARS) [15] emerged as an important research topic in the last years. Recently, entire workshops were devoted to this topic on major conferences (CARS series started in 2009 [16], CAMRa in 2010 [17]).

As we discussed earlier, pre- and post-filtering approaches use traditional recommender algorithms with some kind of filtering or splitting to consider context during learning and/or recommendation. On the other hand contextual modeling focuses on designing algorithms that incorporate context into the model itself. Tensor factorization (TF) follows the contextual modeling flavor of CARS, when contextual information (or simply: context) is incorporated into the recommendation model [12]. TF is a natural extension of matrix factorization for more dimensions, although it is not straightforward how to make it work efficiently.

Let we have a set of items, users and ratings (or events) and assume that additional contexts are available on ratings (e.g. the time of the rating). If we have N_C different contexts we can structure the ratings into a $D = N_C + 2$ dimensional tensor T. The first dimension corresponds to users, the second to items and the subsequent N_C dimensions $[3, \ldots, N_C+2]$ are devoted to context. Note that in order to be able to use this approach, every context dimension must consists of possible context-states that are atomic and categorical values. In other words, the value of a context variable comes from a finite set of atomic values (these values are termed context-states). We want to decompose tensor T into lower rank matrices and/or tensors in a way that the reconstruction of the original tensor from its decomposition approximates the original tensor sufficiently well.

In [18] a sparse HOSVD method is presented that decomposes a D dimensional sparse tensor into D matrices and a D dimensional tensor. The authors use gradient descent on the known ratings to find the decomposition, and by doing so the complexity of one iteration of their algorithm scales *linearly* with the number of non-missing values in the original tensor and *cubically* with the number of features (K). Rendle *et al* proposed a tensor factorization method for tag recommendation [19] that was later adapted to context-aware recommendations [20]. Their model is similar to the one we use to model the relation between users, items and context states. For every entity in every dimension they use two feature vectors and the preference of user u on tag t for item i is approximated by the sum of three scalar products: (1) first user feature vector with first tag vector, (2) second user vector with second item vector and (3) first item vector with second tag vector. The second scalar product can be omitted during recommendations, since it has no effect on the ranking of tags in a given user–item relation, however it can filter noise during the training. The model is generalized to context-aware recommendations by replacing tags by items and items by context. They use gradient descent to minimize the loss function.

Our method also approximates the preferences with the sum of three scalar products, but there are major differences from the previously presented method: (1) there is only one

[1] ITALSx is cited in some of our works as it was described in a closed (publicly non-available) technical report[14] earlier.

feature vector for each entity in our model; (2) our algorithm is able to efficiently handle the implicit feedback case by using computationally effective learning scheme; (3) we use ALS to minimize the error.

The closest to our approach is our previously proposed implicit context aware tensor model [13] that approximates a given cell of the tensor as the sum of the elements in the Hadamard product of three feature vectors (i.e. it uses a full three-way model). iTALS and iTALSx uses the same loss function and optimization procedure, but they use different models and thus require different steps in the learning process. We show how the model differences affect usability in section V.

III. THE iTALSx ALGORITHM

In this section we present our context aware model and an efficient ALS (Alternating Least Squares) based method for training. ALS training fixes all but one feature matrices and computes the columns on the non fixed one by solving a least squares problem for each column.

The presented model uses one context dimension. Problems with several context dimensions can be transformed to use a single dimension by using the Descartes product of the possible context-states of each dimension. Also, the model can be extended to handle arbitrary number of dimensions, by simply adding more dot products to it.

T is a tensor that contains zeroes and ones. The number of ones in T is much lower than the number of zeroes. W contains weights to each element of T. An element of W is greater than 1 if the corresponding element if T is non-zero and 1 otherwise. This approach assumes that the presence of an event is positive preference with a high confidence while its absence is negative preference with a very low confidence. This approach is commonly used for handling implicit feedback [8][13]. In our model we decompose T into three low rank matrices (U, I and C) and approximate a given element by a sum of three dot products. The vectors used in the scalar products are the columns of the matrices corresponding to the given item/user/context state. The following equation describes the preference model:

$$\hat{T}_{u,i,c} = (U_u)^T I_i + (U_u)^T C_c + (I_i)^T C_c \tag{1}$$

During the training of the model we want to minimize the following loss function (weighted RMSE):

$$L(P,Q,C) = \sum_{u=1,i=1,c=1}^{S_U,S_I,S_C} W_{u,i,c}\left(T_{u,i,c} - \hat{T}_{u,i,c}\right)^2 \tag{2}$$

If all but one matrix are fixed (say U and C), L is convex in the non-fixed variables (elements of I in this case). The minimum of L (in I) is reached where its derivative with respect to I is zero. The columns of I can be computed separately because the derivative of L (with respect to I) is linear in I. The derivative for the i^{th} column of I is as follows:

$$\begin{aligned}
\frac{\partial L}{\partial I_i} &= -2\underbrace{\sum_{u=1,c=1}^{S_U,S_C} W_{u,i,c}T_{u,i,c}\left(U_u + C_c\right)}_{\mathcal{O}} + \\
&+2\underbrace{\sum_{u=1,c=1}^{S_U,S_C} W_{u,i,c}\left(U_u + C_c\right)\left(U_u + C_c\right)^T}_{\mathcal{I}} I_i + \\
&+2\underbrace{\sum_{u=1,c=1}^{S_U,S_C} W_{u,i,c}(C_c)^T U_u\left(U_u + C_c\right)}_{\mathcal{B}} = \\
&= -2\mathcal{O} + 2\mathcal{I}I_i + 2\mathcal{B}
\end{aligned} \tag{3}$$

$\mathcal{O}$ can be computed efficiently (see section III-A), but the naive computation of $\mathcal{I}$ and $\mathcal{B}$ is expensive. Therefore these expressions are further transformed by introducing $W'_{u,i,c} = W_{u,i,c} - 1$:

$$\begin{aligned}
\mathcal{I} = &\underbrace{\sum_{u=1,c=1}^{S_U,S_C} W'_{u,i,c}\left(U_u + C_c\right)\left(U_u + C_c\right)^T}_{\mathcal{I}_1} + \\
&+\underbrace{\sum_{u=1,c=1}^{S_U,S_C}\left(U_u + C_c\right)\left(U_u + C_c\right)^T}_{\mathcal{I}_2}
\end{aligned} \tag{4}$$

The sum in $\mathcal{I}_1$ contains at most N^+ non-zero members, because $W'_{u,i,c}$ is zero if the corresponding element is T is zero. Thus its computation is efficient. $\mathcal{I}_2$ is independent of i (it is the same for each column of Q) thus can be precomputed. However its naive computation is still expensive, therefore we further transform $\mathcal{I}_2$ as follows:

$$\begin{aligned}
\mathcal{I}_2 = S_C &\underbrace{\sum_{u=1}^{S_U} U_u\left(U_u\right)^T}_{\mathcal{M}^{(U)}} + S_U \underbrace{\sum_{c=1}^{S_C} C_c\left(C_c\right)^T}_{\mathcal{M}^{(C)}} + \\
+ &\underbrace{\left(\sum_{c=1}^{S_C} C_c\right)}_{\mathcal{X}^{(C)}}\underbrace{\left(\sum_{u=1}^{S_U} U_u\right)^T}_{(\mathcal{X}^{(U)})^T} + \underbrace{\left(\sum_{u=1}^{S_U} U_u\right)}_{\mathcal{X}^{(U)}}\underbrace{\left(\sum_{c=1}^{S_C} C_c\right)^T}_{(\mathcal{X}^{(C)})^T} = \\
= &S_C\mathcal{M}^{(U)} + S_U\mathcal{M}^{(C)} + \mathcal{X}^{(C)}(\mathcal{X}^{(U)})^T + \mathcal{X}^{(U)}(\mathcal{X}^{(C)})^T
\end{aligned} \tag{5}$$

The members ($\mathcal{M}^{(U)}$, $\mathcal{M}^{(C)}$, $\mathcal{X}^{(U)}$, $\mathcal{X}^{(C)}$) in equation 7 can be computed efficiently. Note that the recomputation of $\mathcal{M}^{(U)}$ and $\mathcal{X}^{(U)}$ is only necessary when U changes and therefore we include the cost of recomputing these variables to the cost of recomputing U. We can perform similar steps for $\mathcal{B}$, separating it into two parts, one of which can be rewritten using the variables above. The decomposition is shown in the following equation:

$$\begin{aligned}\mathcal{B} = &\underbrace{\sum_{u=1,c=1}^{S_U,S_C} W'_{u,i,c}(C_c)^T U_u \left(U_u + C_c\right) +}_{\mathcal{B}_1} \\ &+\mathcal{M}^{(U)}\mathcal{X}^{(C)} + \mathcal{M}^{(C)}\mathcal{X}^{(U)}\end{aligned} \tag{6}$$

Now all we have to do in order compute the desired column of I is to solve a $K \times K$ system of linear equations:

$$I_i = \left(\mathcal{I}_2 + \mathcal{I}_1\right)^{-1}\left(\mathcal{O} - \left(\mathcal{B}_2 + \mathcal{B}_1\right)\right) \tag{7}$$

Bias and regularization can be easily added to the method, thus they are omitted in this deduction for the sake of clearer presentation.

Algorithm III.1 presents the pseudocode for training the model, that is the straight translation of the deduction above. The method is named iTALSx.

Algorithm III.1 iTALSx algorithm

Input: T: $S_1 \times S_2 \times S_3$ sized tensor of zeroes and ones; W: $S_1 \times S_2 \times S_3$ sized tensor containing the weights; K: number of features; E: number of epochs; $\{\lambda_m\}_{m=1,2,3}$: regularization parameters

Output: $\{M^{(i)}\}_{m=1,2,3}$ $K \times S_i$ sized matrices

Note: $S_1 = S_U$, $S_2 = S_I$, $S_3 = S_C$, $M^{(1)} = U$, $M^{(2)} = I$, $M^{(3)} = C$

```
procedure iTALSx($T, W, K, E$)
for $m = 1, \ldots, 3$ do
    $M^{(m)} \leftarrow$ Random $K \times S_m$ sized matrix
    $\mathcal{M}^{(m)} \leftarrow \sum_{j=1}^{S_m} M_j^{(m)}(M_j^{(m)})^T$
    $\mathcal{X}^{(m)} \leftarrow \sum_{j=1}^{S_m} M_j^{(m)}$
end for
for $e = 1, \ldots, E$ do
    for $m = 1 \ldots, 3$ do
        $p \leftarrow (m-1)\%3$
        $n \leftarrow (m+1)\%3$
        $\mathcal{I}_2 \leftarrow \mathcal{M}^{(p)}S_n + \mathcal{M}^{(n)}S_p + \mathcal{X}^{(p)}(\mathcal{X}^{(n)})^T + \mathcal{X}^{(n)}(\mathcal{X}^{(p)})^T$
        $\mathcal{B}_2 \leftarrow \mathcal{M}^{(p)}\mathcal{X}^{(p)} + \mathcal{M}^{(n)}\mathcal{X}^{(p)}$
        for $i = 1..S_m$ do
            $\mathcal{I} \leftarrow \mathcal{I}_2$
            $\mathcal{O} \leftarrow 0$
            $\mathcal{B} \leftarrow \mathcal{B}_2$
            for all $\{t | t = T_{j_1,j_2,j_3}, j_m = i, t \neq 0\}$ do
                $w \leftarrow$ corresponding value in $W$ to $t$ in $T$
                $v \leftarrow M^{(p)} + M^{(n)}$
                $\mathcal{I} \leftarrow \mathcal{I} + wvv^T$
                $\mathcal{O} \leftarrow \mathcal{O} + wtv$
                $\mathcal{B} \leftarrow \mathcal{B} + wM^{(p)}(M^{(n)})^T v$
            end for
            $M_i^{(m)} \leftarrow (\mathcal{I} + \lambda_m I)^{-1}(\mathcal{O} - \mathcal{B})$
        end for
        $\mathcal{M}^{(m)} \leftarrow \sum_{j=1}^{S_m} M_j^{(m)}(M_j^{(m)})^T$
        $\mathcal{X}^{(m)} \leftarrow \sum_{j=1}^{S_m} M_j^{(m)}$
    end for
end for
return $\{M^{(m)}\}_{m=1\ldots3}$
end procedure
```

A. Complexity

The complexity of one epoch (i.e. computing each matrix once) is $O\left(K^3(S_U + S_I + S_C) + K^2 N^+\right)$, thus it scales linearly with the number of non-zeros in the tensor and cubically with the number of factors. Since in practical problems $N^+ \gg S_U + S_I + S_C$, the scaling of the method is practically quadratical in K when small $(10 \ldots 400)$ K values are used (also common in practice).[2]

The complexity of recomputing I (as in the deduction above) is $O(K^3 S_I + K^2 N^+)$. This complexity also contains the recomputation of $\mathcal{M}^{(I)}$ and $\mathcal{X}^{(I)}$ that are needed later for the computation of the other two matrices. The aforementioned complexity consists of calculating

- (1) $\mathcal{O}$ for each column in equation (3) in $O(KN^+)$ time as only N_i^+ elements of T are non-zeroes for i^{th} item ($N^+ = \sum_{i=1}^{M} N_i^+$);
- (2) $\mathcal{I}_1$ for each column in equation (4) in $O(K^2 N^+)$ time as $W'_{u,i,c} = (W_{u,i,c} - 1)$ is zero if the value of $T_{u,i,c}$ is zero;
- (3) $\mathcal{I}_2$ in equation (7) in $O(K^2)$ time from the precomputed values $\mathcal{M}^{(U)}$, $\mathcal{M}^{(C)}$, $\mathcal{X}^{(U)}$, $\mathcal{X}^{(C)}$;
- (4) $\mathcal{B}$ in equation (6) in $O(KN^+ + K^2)$ time analogously to the computation of $\mathcal{I} = \mathcal{I}_1 + \mathcal{I}_2$;
- (5) solving the systems of equations for all columns in $O(S_I K^3)$ time;
- (6) recomputing $\mathcal{M}^{(I)}$ and $\mathcal{X}^{(I)}$ based on the new I matrix in $O(S_I K^2)$ time

IV. Comparison with iTALS

In earlier work we recently proposed a context aware tensor model (coined iTALS) for the implicit feedback problem. This model is a full three-way model that approximates the elements in T with the sum of the elements in the Hadamard products of three vectors. Mathematically the model of iTALS [13] contains the iTALSx model as:

$$\begin{aligned}\hat{T}_{u,i,c}^{\text{iTALS}} =& 1^T \left(U_u \circ I_i \circ C_c\right) \\ 3 \cdot \hat{T}_{u,i,c}^{\text{iTALS}} =& 1^T \left(U_u \circ I_i \circ C_c\right) + 1^T \left(U_u \circ I_i \circ C_c\right) + \\ &+ 1^T \left(U_u \circ I_i \circ C_c\right) \\ \hat{T}_{u,i,c}^{\text{iTALSx}} =& 1^T \left(I_i \circ C_c\right) + 1^T \left(U_u \circ C_c\right) + 1^T \left(U_u \circ I_i\right) \\ \hat{T}_{u,i,c}^{\text{iTALSx}} =& 1^T \left(1 \circ I_i \circ C_c\right) + 1^T \left(U_u \circ 1 \circ C_c\right) + \\ &+ 1^T \left(U_u \circ I_i \circ 1\right),\end{aligned} \tag{8}$$

where $\circ$ denotes the Hadamard product of the argument vectors.

It is more interesting to compare the models from the recommendation aspect. The main goal of a collaborative

[2] This can be reduced to a theoretically quadratic and practically linear scaling by applying approximate least squares solvers like conjugate gradient instead of the exact solver, like in [21].

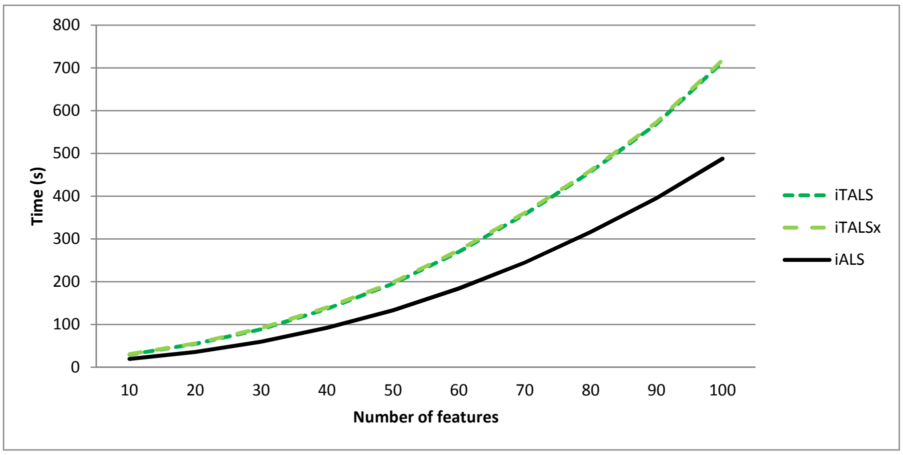


Fig. 1. The time of one epoch (computing each feature matrix once) with iTALS and iTALSx using different number of features. (Measurements on the VoD data set; only one core used.) Results for iALS are also depicted.

filtering algorithm is to learn the user–item relations (e.g. which user likes which item). iTALS adds context to the model and approximates the user–item relation in the 3 dimensional space. It reweights the user–item relations by a context state dependent vector, which becomes more accurate with more factors [13]. On the other hand, iTALSx uses a composite model and approximates the user–item relation by the sum of approximations in user–item, user–context and item–context sub-spaces, where the feature vectors in the sub-spaces are constrained by requiring a single feature vector for each entity. Consequently, the descriptive power of iTALS is larger, which can be however only leveraged at a sufficiently fine resolution of the feature space, requiring many factors. At low factor models, the boundaries of different characteristics is blurred by reweighting and the model becomes less precise. In such cases, iTALSx is expected to be more accurate, since the sub-space models can be learnt easier.

A. Complexity and training times

The complexity is $O(N^+K^2 + (S_U + S_I + S_C)K^3)$ for both iTALS and iTALSx. Since in practice $N^+ \gg (S_U + S_I + S_C)$, each method scales with K^2 when low-factor models are used. However the training time of iTALSx is slightly higher, because (a) iTALS does not require $\mathcal{X}^{(m)}$ for its computations; (b) the computations in iTALSx require a few extra operations (see Figure 1).

Figure 1 also contains the training times for non-context-aware (2D) iALS algorithm, that uses a similar method for learning. The complexity of iALS is $O(N^+K^2 + (S_U + S_I)K^3)$. This means that the running times of iALS and the context-aware methods differ only in a constant multiplier, that is proportional to the number of matrices to be recomputed (see Figure 1), but the time to compute one feature matrix is virtually the same for these algorithms.

V. RESULTS

We used five data sets to evaluate our algorithm. Two of them (Grocery and VoD) are proprietary data sets and contain real-life implicit data. The other three data sets (LastFM [22], TV1 and TV2 [23]) are publicly available, but might have been transformed/cleaned prior release. The properties of the data sets are summarized in Table I. The column “Multi” shows the average multiplicity of user-item pairs in the training events.[3] Data density is measured without context, with seasonality (-S) and with sequentiality (-Q). The first event in the test data is after the last event of the training data. The length of the test period was selected to be at least one day, and depends on the domain and the frequency of events. We used the artists as items in LastFM.

The evaluation metric used here is recall@N. Recall is the proportion of the number of recommended and relevant items to the number of relevant items. Item i is considered relevant for user u if the user has at least one event for that item in the test set. The item is recommended at cut-off N if it belongs to the topN of the recommendation list[4]. We chose cut-off 20 because the length of the recommendation list is limited as users are exposed to a few recommended items at a time. The evaluation is event based, meaning that if the user has multiple events on an item in the test set then that item is considered multiple times.

Recall@N suites the recommendation task really well from a practical point of view. During a session the user is exposed to some recommended items (e.g. a few on each page visited) and the recommendation is successful if she interacts (e.g. buys, watches) with these items. The items further down the recommendation list are irrelevant, because they won’t be shown to the user. 20 as cut-off is a fair estimation of the items the user sees during a session (e.g. 4 pages visited, 5 recommendations per page). In most practical settings the order of the topN items is irrelevant due to the placement of recommendations on the site.[5]

[3] This value is 1.0 at two data sets: TV1 and TV2 due to possible filtering of duplicate events.

[4] The recommendation list is generated by ordering the items for a user (under the given context) by their predicted preference values in descending order.

TABLE I
MAIN PROPERTIES OF THE DATA SETS

Dataset	Domain	Training set							Test set	
		#Users	#Items	#Events	Density	Density-S	Density-Q	Multi	#Events	Length
Grocery	E-grocery	24947	16883	6238269	0.61%	0.15%	9.37E-5%	3.0279	56449	1 month
TV1	IPTV	70771	773	544947	1.02%	0.17%	1.63E-3%	1.0000	12296	1 week
TV2	IPTV	449684	3398	2528215	0.17%	0.028%	6.42E-5%	1.0000	21866	1 day
LastFM	Music	992	174091	18908597	0.52%	0.21%	2.31E-5%	21.2715	17941	1 day
VoD	IPTV	480016	46745	22515406	0.25%	0.046%	1.91E-5%	1.2135	1084297	1 day

We experimented with two types of context. The first is seasonality. It consists of a season (or periodicity) and time bands therein. Each event is assigned to a time band based on its time stamp. The idea behind seasonality is that people have daily/weekly/yearly routines and those are different for different types of people (same goes for the items' consumption patterns). As season, we define *one week* for Grocery (as people usually go shopping once a week) and *one day* for the other data sets (as movie and music consumption shows daily periodicity). The days of the week were used for Grocery and four hour periods within the day for the other data sets as time bands.

The second type of context is sequentiality [13]. The context state of an event is the previously consumed item of the same user. This context enables distinction between item groups with different repetitiveness patterns (e.g. item groups that can and those that should not be recommended subsequently). Sequentiality tackles this problem through the co-occurrence of the items. It – implicitly – also serves weak information on the user or her properties (e.g.: mood) if the subsequent events are close to each other. Each context state corresponds to a singular preceding item.

iTALSx is compared mainly to iTALS in order to find the differences between the behavior of the pairwise model and the three-way model. Results for the non-context-aware iALS [8] are also presented as a baseline. The number of features was set to 20, 40 and 80, the number of epochs was 10. These are typical settings in real life environments. Other parameters such as regularization coefficients were optimized on a hold-out set of the training data, then the algorithm was retrained with the optimal parameters on the whole training data.

Table II contains the results. Measurements with seasonality and sequentiality are denoted with the -S and -Q postfix respectively. As expected, context improves recommendation accuracy. There are two contradictory examples where the context-unaware method performs significantly better than iTALS. This is due to sensitivity of the elementwise model to noise and the poor quality of this seasonal context for the TV2 dataset and the outstanding sparsity of TV2 dataset compared to the others in this setting. The range of improvement for iTALS and iTALSx over the context-unaware baseline is $11\% - 53\%$ and $7\% - 63\%$ respectively, with seasonality; $7\% - 248\%$ and $11\% - 274\%$ with sequentiality. I.e. iTALSx increases the accuracy slightly more than iTALS.

TABLE II
RECALL@20 FOR iTALS, iTALSx AND iALS. MEASUREMENTS WITH SEASONALITY AND SEQUENTIALITY ARE DENOTED WITH THE -S AND -Q POSTFIX RESPECTIVELY.

GROCERY					
K	iALS	iTALS-S	iTALSx-S	iTALS-Q	iTALSx-Q
20	0.0649	0.0990	0.1027	0.1220	0.1182
40	0.0714	0.1071	0.1164	0.1339	0.1299
80	0.0861	0.1146	0.1406	0.1439	0.1431
TV1					
K	iALS	iTALS-S	iTALSx-S	iTALS-Q	iTALSx-Q
20	0.1189	0.1167	0.1248	0.1417	0.1524
40	0.1111	0.1235	0.1127	0.1515	0.1417
80	0.0926	0.1167	0.0942	0.1553	0.1295
TV2					
K	iALS	iTALS-S	iTALSx-S	iTALS-Q	iTALSx-Q
20	0.2162	0.1734	0.2220	0.2322	0.2393
40	0.2161	0.2001	0.2312	0.3103	0.2866
80	0.2145	0.2123	0.2223	0.2957	0.3006
LASTFM					
K	iALS	iTALS-S	iTALSx-S	iTALS-Q	iTALSx-Q
20	0.0448	0.0674	0.0503	0.1556	0.1675
40	0.0623	0.0888	0.0599	0.1657	0.1869
80	0.0922	0.1290	0.0928	0.1864	0.1984
VOD					
K	iALS	iTALS-S	iTALSx-S	iTALS-Q	iTALSx-Q
20	0.0633	0.0778	0.0790	0.1039	0.0821
40	0.0758	0.0909	0.0916	0.1380	0.1068
80	0.0884	0.0996	0.0990	0.1723	0.1342

The better between iTALS and iTALSx in the same setting (i.e.: same context, number of features, dataset) is highlighted by a light gray background. Generally, iTALSx performs better if the number of features is lower. Also, there seems to be a loose connection between the density of the dataset and the relative accuracy of the two models. With a given context, iTALSx performs better if the density of the dataset is lower. High sparsity (lower density) is a common property of real life datasets, therefore iTALSx is beneficial for practical applications.

Figure 2 compares iTALS and iTALSx using high number of features on the LastFM dataset. With seasonality, iTALS is already better than iTALSx, even with 40 features. The recommendation accuracy of iTALS improves faster as the number of features increases. With sequentiality, iTALSx starts off with significantly better accuracy, but as the number of

[5] This does not apply if some items are highlighted from the recommendations, e.g. the picture for the first recommended item is larger.

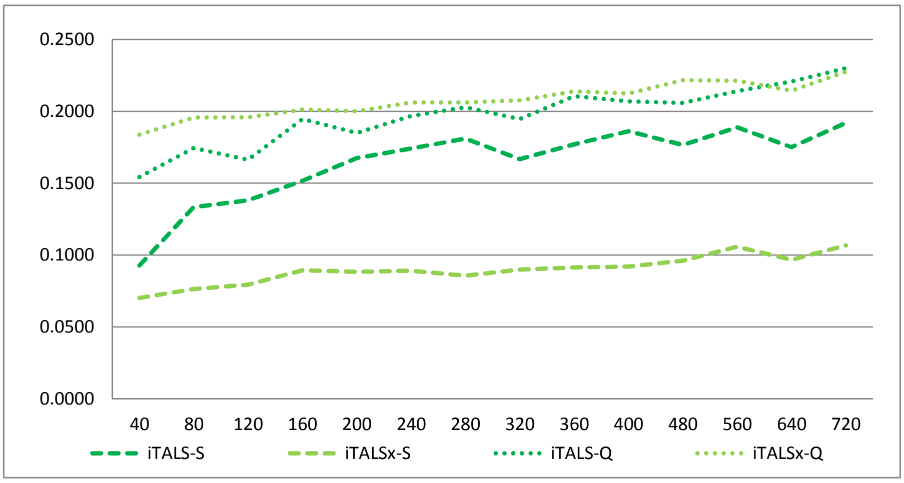


Fig. 2. Recall@20 values for iTALS and iTALSx with seasonality (-S) and sequentiality (-Q) with the number of features ranging from 40 to 720 on the LastFM dataset.

features increase, the difference becomes less significant and it disappears at high factor models. The speed of accuracy improvement is better for iTALS in both cases.

The blurring effect of the low feature models makes learning difficult for iTALS, especially if the dataset is sparse. Sparser datasets are generally more noisy, and the elementwise model is more sensitive to noise by nature, because of the reweighting of the user–item relation in that model. Our assumption about the learning capabilities of the algorithms and their connection to the finer representation of entities are underpinned as iTALS can outperform iTALSx when the number of features is sufficiently large or if the dataset is more dense. These results imply that one should use iTALSx when the dataset is sparse and we can not afford high feature models (that is most common in practical applications).

VI. Conclusion

In this paper we presented iTALSx, an efficient context-aware factorization method for implicit feedback data, which approximates preferences as the sum of three scalar products. It scales cubically (quadratically in practice) with the number of features (K) and linearly with the number of events. We compared it to iTALS, a similar method that uses a different (three-way) model. We found that both models have their advantages. The pairwise model of iTALSx is more efficient in terms of accuracy if the number of features is low and the dataset is more sparse. This is a usual setting in real life problems. However if one can afford high factor models or the data is more dense, iTALS should be used, as its learning capability is higher. Thus it can achieve better results if the number of features is sufficient.

Acknowledgements

The research leading to these results was partially performed in the CrowdRec project, which has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement n° 610594.

References

[1] P. Lops, M. Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender Systems Handbook*. Springer, 2011, pp. 73–105.

[2] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in Artificial Intelligence*, pp. Article ID 421 425 (1–19), 2009.

[3] I. Pilászy and D. Tikk, "Recommending new movies: Even a few ratings are more valuable than metadata," in *Recsys'09: ACM Conf. on Recommender Systems*, 2009, pp. 93–100.

[4] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender Systems Handbook*, F. Ricci *et al.*, Eds. Springer, 2011, pp. 145–186.

[5] R. Bell and Y. Koren, "Scalable collaborative filtering with jointly derived neighborhood interpolation weights," in *ICDM'07: IEEE Int. Conf. on Data Mining*, 2007, pp. 43–52.

[6] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Major components of the Gravity recommendation system," *SIGKDD Explor. Newsl.*, vol. 9, pp. 80–83, December 2007.

[7] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Advances in Neural Information Processing Systems 20*. MIT Press, 2008.

[8] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *ICDM-08: IEEE Int. Conf. on Data Mining*, 2008, pp. 263–272.

[9] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*. Springer US, 2011, pp. 1–35.

[10] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, and A. Hanjalic, "Climf: learning to maximize reciprocal rank with collaborative less-is-more filtering," in *Proceedings of the sixth ACM conference on Recommender systems*, ser. RecSys '12. ACM, 2012, pp. 139–146.

[11] G. Takács and D. Tikk, "Alternating least squares for personalized ranking," in *Proceedings of the sixth ACM conference on Recommender systems*, ser. RecSys '12. ACM, 2012, pp. 83–90.

[12] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recsys'08: ACM Conf. on Recommender Systems*, 2008, pp. 335–336.

[13] B. Hidasi and D. Tikk, "Fast ALS-based tensor factorization for context-aware recommendation from implicit feedback," in *Proc. of the ECML-PKDD, Part II*, ser. LNCS. Springer, 2012, no. 7524, pp. 67–82.

[14] B. Hidasi, "Technical report on iTALSx," Gravity R&D Inc., Tech. Report Series 2012-2, 2012.

[15] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, "Incorporating contextual information in recommender systems using a multidimensional approach," *ACM Trans. Inf. Syst.*, vol. 23, no. 1, pp. 103–145, 2005.

[16] G. Adomavicius and F. Ricci, "Workshop on context-aware recommender systems (CARS-2009)," in *Recsys'09: ACM Conf. on Recommender Systems*, 2009, pp. 423–424.

[17] A. Said, S. Berkovsky, and E. W. D. Luca, "Putting things in context: Challenge on context-aware movie recommendation," in *CAMRa'10: Workshop on Context-Aware Movie Recommendation*, 2010, pp. 2–6.

[18] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: N-dimensional tensor factorization for context-aware collaborative filtering," in *Recsys'10: ACM Conf. on Recommender Systems*, 2010, pp. 79–86.

[19] S. Rendle and L. Schmidt-Thieme, "Pairwise interaction tensor factorization for personalized tag recommendation," in *WSDM'10: ACM Int. Conf. on Web Search and Data Mining*, 2010, pp. 81–90.

[20] Z. Gantner, S. Rendle, and L. Schmidt-Thieme, "Factorization models for context-/time-aware movie recommendations," in *Proc. of the Workshop on Context-Aware Movie Recommendation*, 2010, pp. 14–19.

[21] B. Hidasi and D. Tikk, "Context-aware recommendations from implicit data via scalable tensor factorization," *ArXiv e-prints*, 2013.

[22] O. Celma, *Music Recommendation and Discovery in the Long Tail*. Springer, 2010.

[23] P. Cremonesi and R. Turrin, "Analysis of cold-start recommendations in IPTV systems," in *Proc. of the 2009 ACM Conference on Recommender Systems*, 2009.

Balázs Hidasi is a datamining researcher. His research interest cover a broad spectrum of machine learning / data mining algorithms and problems. His recent research revolves around recommender algorithms for real life recommendation problems, that include research related to implicit feedback, context-awareness, hybrid collaborative filtering and so on. Before that he worked on time series classification. Since 2010 he is employed by Gravity Research and Development Inc., a recommendation service provider company.

There he carries out his research and applies the resulting algorithms directly to real life problems. He graduated with highest honors from the Budapest University of Technology, and received his masters degree in computer science and engineering in 2011.

Our reviewers in 2014

The quality of a research journal depends largely on its reviewing process and, first of all, on the professional service of its reviewers. It is my pleasure to publish the list of our reviewers in 2010 and would like to express my gratitude to them for their devoted work.

Your Editor-in-Chief

Imre Abos,
BME, Hungary

Péter Almási,
University of Debrecen, Hungary

Luigi Atzori,
University of Cagliari

Simon Back,
Salzburg University of Applied Sciences

Péter Bakonyi,
BME, Hungary

János Botzheim,
Széchenyi István University, Hungary

Tibor Cinkler,
BME, Hungary

Tibor Csendes,
Szeged University of Sciences, Hungary

László Czuni,
Pannon University, Hungary

Virgil Dobrota,
Technical University of Cluj-Napoca, Romania

Péter Ekler,
BME, Hungary

István Frigyes,
BME, Hungary

Péter Fülöp,
Ericsson Research, Hungary

Christian Gütl,
University of Graz, Austria

László Gyöngyösi,
BME, Hungary

Khairi Hamdi,
University of Manchester, UK

Thomas Heistrachter,
Salzburg University of Applied Sciences, Austria

Bálint Horváth,
BME, Hungary

Jukka Huhtamakki,
Tampere University of Technology, Finland

Árpád Huszák,
BME, Hungary

Márton Ispány,
University of Debrecen, Hungary

Péter Jeszenszky,
University of Debrecen, Hungary

Péter Kántor,
BME, Hungary

László Kóczy,
Széchenyi István University, Hungary

János Kormos,
University of Debrecen, Hungary

László Lakatos,
Eötvös Loránd University of Sciences, Hungary

Václav Matyas,
Masaryk University, Brno, Czech Republic

Miklos Molnar,
University of Montpellier, France

Sándor Molnár,
BME, Hungary

Albert Mráz,
BME, Hungary

Francesco De Natale,
University of Trento, Italy

Péter Orosz,
University of Debrecen, Hungary

Gábor Rétvári,
BME, Hungary

György Strausz,
BME, Hungary

János Sztrik,
University of Debrecen, Hungary

Do Van Tien,
BME, Hungary

Nándor Vannai,
BME, Hungary

Pál Varga,
BME, Hungary

Mukundan Venkataraman,
University of Central Florida, USA

György Wersényi,
Széchenyi István University, Hungary

(* BME – Budapest University of Technology and Economics)

Contents

of the Infocommunications Journal 2014 (Volume VI)

2014/1 *Infocommunications Journal*

2014/2 *Infocommunications Journal*

2014/3 *Infocommunications Journal*

2014/4 *Infocommunications Journal*

SCIENTIFIC ASSOCIATION FOR INFOCOMMUNICATIONS

FOUNDED IN 1949

SCIENTIFIC ASSOCIATION FOR INFOCOMMUNICATIONS HUNGARY

Who we are

Founded in 1949, the Scientific Association for Infocommunications (formerly known as Scientific Society for Telecommunications) is a voluntary and autonomous professional society of engineers and economists, researchers and businessmen, managers and educational, regulatory and other professionals working in the fields of telecommunications, broadcasting, electronics, information and media technologies in Hungary.

Besides its more than 1300 individual members, the Scientific Association for Infocommunications (in Hungarian: Hírközlési és Informatikai Tudományos Egyesület, HTE) has more than 60 corporate members as well. Among them there are large companies and small-and-medium enterprises with industrial, trade, service-providing, research and development activities, as well as educational institutions and research centers.

HTE is a Sister Society of the Institute of Electrical and Electronics Engineers, Inc. (IEEE) and the IEEE Communications Society. HTE is corporate member of International Telecommunications Society (ITS).

What we do

HTE has a broad range of activities that aim to promote the convergence of information and communication technologies and the deployment of synergic applications and services, to broaden the knowledge and skills of our members, to facilitate the exchange of ideas and experiences, as well as to integrate and harmonize the professional opinions and standpoints derived from various group interests and market dynamics.

To achieve these goals, we…

- contribute to the analysis of technical, economic, and social questions related to our field of competence, and forward the synthesized opinion of our experts to scientific, legislative, industrial and educational organizations and institutions;
- follow the national and international trends and results related to our field of competence, foster the professional and business relations between foreign and Hungarian companies and institutes;
- organize an extensive range of lectures, seminars, debates, conferences, exhibitions, company presentations, and club events in order to transfer and deploy scientific, technical and economic knowledge and skills;
- promote professional secondary and higher education and take active part in the development of professional education, teaching and training;
- establish and maintain relations with other domestic and foreign fellow associations, IEEE sister societies;
- award prizes for outstanding scientific, educational, managerial, commercial and/or societal activities and achievements in the fields of infocommunication.

Contact information

President: **DR. GÁBOR HUSZTY** • *ghuszty@entel.hu*

Secretary-General: **DR. ISTVÁN BARTOLITS** • *bartolits@nmhh.hu*

Managing Director, Deputy Secretary-General: **PÉTER NAGY** • *nagy.peter@hte.hu*

International Affairs: **ROLLAND VIDA, PhD** • *vida@tmit.bme.hu*

Addresses

Office: H-1051 Budapest, Bajcsy-Zsilinszky str. 12, Room: 502
Phone: +36 1 353 1027, Fax: +36 1 353 0451
E-mail: *info@hte.hu*, Web: *www.hte.hu*