*Electronic Notes in Theoretical Computer Science 50 No. 1 (2001) { Proc. ATMOS 2001 URL:* [*http://www.elsevier.nl/locate/entcs/volume50.html*](http://www.elsevier.nl/locate/entcs/volume50.html) *11 pages*

*Web-Based Integrated Timetable Information* System for Railways and Airlines

# *Krzysztof Goczyla 1*

*Department of Applied Informatics Technical University of Gdansk Gdansk, Poland*

*Abstract*

*The paper presents a traÆc information system called KRJ that covers Polish railways and airlines transportation systems. Basic algorithmic foundations for the system are given. The system was previously designed as a standalone program. Recently, it has been moved to Internet and made available at a commercial Web site as a service called Ekspres. The process of migration into Internet service and problems encountered are described. The paper concludes with experiences gained so far from a one-year exploitation under operational workload.*

# *1 Introduction*

*The paper presents a timetable information system developed for Polish Rail-* ways (PKP) and Polish Airlines (LOT). This system has been known in Poland as KRJ. The system was previously (i.e. in 1992) intended to work only in o -line mode, in DOS and Windows operating environments. In 2000, a joint e ort was made to move the system to Internet. The implementation of this idea required a set of complex programming and organisational tasks. In the paper we describe this implementation and the result { an on-line system called Ekspres, which has been operating in one of the largest Polish Web sites called \Wirtualna Polska" (in English: \Virtual Poland") [1].

*To make the presentation clear, in Section 2 we present brie y main algo-* rithmic foundations for KRJ. Theoretical and some implementation issues of the o -line version of KRJ has already been presented (see [3], [4], [5], [6], [7], [8]); here we focus on the integration issues that enable the system to include di erent means of transport, with their peculiarities, in one search engine. In Section 3 we present implementation tasks that had to be performed in order to move the system from o -line environments to an on-line environment and

*1 Email:* [*kris@pg.gda.pl*](mailto:kris@pg.gda.pl)

*c 2001 Published by Elsevier Science B. V. Open access under* [*CC BY-NC-ND license.*](http://creativecommons.org/licenses/by-nc-nd/3.0/)

*the resulting architecture of the Web version of the system. Section 4 discusses* some user interface issues related to that kind of information systems. Section

*5 concludes the paper with experiences gained so far from exploitation of the* system at a commercial Web site.

# *2 A model of a transportation network*

*In a model applied in the KRJ search engine, we consider each means of* transport as belonging to one of two classes. The rst class is composed of means of transport that have a xed timetable (like trains, planes, or buses). The second class is composed of means that do not have such a timetable (i.e. you can start your travel practically at any moment). Private cars, bicycles or one's feet are examples of such means of transport. You can also include into this class some periodic means of transport with high frequency of circulating, like underground at rush hours. Let us note that short walks on foot usually are very important components of travel. They occur either \implicitly" (when you must change a platform at a train station), or \explicitly", when you must change a means of transport and reach a closely situated bus station after

*nishing travel by train.*

*We model a transportation network as a weighted directed multigraph.* The idea behind this approach is to apply algorithms for nding shortest paths, known from the graph theory. In our model, there are arcs of two types, ac- cording to the two classes of means of transport. The discrete arcs correspond to the passages covered by the means of transport with a timetable. The free arcs correspond to passages covered by the means without a xed timetable and to changes of means of transport (e.g. changing trains or changing from a train to a bus). Let us make it more precise:

*De nition 2.1 The discrete arc is an arc of a directed multigraph for which* there is given a limit time TD. The weight of a discrete arc wij for a given start time t is de ned as:

*(1)*

*wij*

*= 8< w if t TD*

*1 if t > T D*

*:*

*where w is a constant real value, w 2 R+.*

*De nition 2.2 The free arc is an arc of a directed multigraph for which there* is not given any limit time. The weight of a free arc is a continuous function of t, the time of leaving the starting node of the arc. The function depends on the means of transport.

*De nition 2.3 The ride is a segment of travel covered by one vehicle accord-* ing to its timetable (if any).

*Figure 1 depicts a fragment of the graph model, in which Xi models a* station for a means of transport with timetable and Xm models a station for

*a means of transport without timetable. It is assumed that a walk on foot is* necessary between these two stations.



vd

l,1,2

X l

X j

va

vl,1,1

d

l,1

X i

vd

j,3,1

va

j,5

va

i,1

va

i,2

vd

i,1,1

vd

vd

i,4,2

i,1,2

vd

i,4,1

vd

i,5,1

vi,3,2

d

X

v d

k

X

v

d

i,2,2

m

i,5,2 d

vi,2,1

vd

i,3,1

va

k, 2

va

vd

k, 1

m,1,2

vd

m,1,1

va

k, 3,1

va

k, 3,2

Arcs inside *xi* :

 direct connection; weight = 0  connection with a change; weight = *p*

Arcs between *xi* and other stations:

 a discrete arc; its weight depends on departure and arrival time and on the means of transport

 a free arc; its weight depends on the duration of passage and on the means of transport

*Fig. 1. A fragment of a transportation network model*

*V is the set of nodes with the following properties:*

*For each arrival in a single ride there is created a separate node of the graph,* and for each departure in a single ride there is created a set of nodes of the

*; v*

*i;m*

*graph. In other words, each station Xi is modelled by nodes va*

*i;1*

*a i;2*

*;::: ; va*

*corresponding to arrivals of a vehicle, and by nodes vd*

*i;1;1*

*d*

*i;2;1*

*; v*

*;::: ; vd*

*m*

*i;n ;m*

*corresponding to departures of a vehicle (we use the following notation: if*

*an arrival node is va , then the consecutive departure nodes for this arrival*

*i;j*

*node are vd*

*i;1;j*

*d*

*i;2;j*

*; v*

*, etc.).*

*For the rides covered by a vehicle without a timetable there are sets of nodes* for both: arrivals and departures.

*The model is a multigraph G(V; E; W ). E is the set of arcs with the* following properties:

*Within a single station, there is a free arc from a node representing the* arrival in a single ride to each node representing any departure in the same ride. This arc corresponds to a change at the station or going through the station without a change.

*For each ride there is a cluster of discrete arcs directed from the departure* nodes to a common arrival node (i.e. having the same ending node).

*For walks between the stations there are free arcs. Single arcs can replace*

*; v*

*d*

*the clusters of the arcs of this type. In Figure 1 these arcs are: (vd*

*i;5;1*

*; v*

*d*

*m;1;1)*

*d*

*and (v*

*i;5;2*

*m;1;2).*

*For rides covered by a vehicle without a timetable there are free arcs (again,*

*; v*

*a*

*single arcs can replace the clusters). In Figure 1 these arcs are: (vd*

*m;1;1*

*; v*

*a*

*k;3;1)*

*d*

*and (v*

*m;1;2*

*k;3;2).*

*W is a weight function that depends on time and has the following prop-* erties:

*The weight of a discrete arc corresponding to one ride is determined by the* time of arrival to the ending node (or, if we are nding the latest time of departure instead of the earliest time of arrival, by the time of departure from the starting node). The weight may also include some other, non-time characteristics.

*The weight of a free arc corresponding to the passage covered by a single* vehicle is equal to zero.

*The weight of a free arc corresponding to a change of a vehicle is equal to* p, which is referred to as the time equivalent to one change.

*The weight of a free arc corresponding to a passage covered bya vehicle with* no xed timetable is speci ed as a function that depends on the departure time and on the type of means of transport. This function may di er among the arcs.

*To nd an optimal connection it is necessary to take into account many dif-* ferent variants of reaching all the intermediate nodes for the connection. While

*nding connections, using e.g. an algorithm based on Dijkstra's algorithm or* one of its numerous variants ([2], [10]), from the source to the destination, two values are evaluated in every node: the current time of entering the node and the current weight of the connection. Due to the model presented above, the weights of the paths are appropriately compared in the nodes visited. It is also possible to include the following factors into the weights of the paths: the time of arrival, the number of changes and the time of departure (as late as possible).

*In the process of nding a connection from a given source node to a* given destination node, the algorithm starts simultaneously from all the nodes

*d*

*v*

*p;1;1*

*d*

*p;2;1*

*; v*

*;::: ; vd*

*that model departures from the source node, and ends*

*by reaching any of the nodes that model arrivals to the destination node (i.e.*

*p;m;1*

*k;m*

*nodes va*

*k;1*

*a k;2*

*;::: ; va*

*). There is however a remarkable di erence in handling*

*the rides with a timetable and the rides with no timetable.*

*; v*

*For a single ride with a xed timetable there is only one common ending* node. One can show that the algorithm that nds a connection can store such a ride only once. Because for such a ride there is one xed departure time and one xed arrival time, these times can be used directly to evaluate the current time and the weight of the connection.

*For each passage covered by a means of transport without any xed time-* table there is an arc with separate starting and ending nodes. It is due to the fact that the times of arrivals at the ending nodes are di erent. If after reaching the node Xk the travel is to be extended further using a means of transport with a xed timetable, the possibilities of extending the travel and the weights obtained will depend on the time of reaching the node Xk.

*The proper choice of an algorithm to nd a minimal-cost path in such a* weighted directed graph is a ected by the graph density (the average number of arcs per node, see e.g. the discussion in [10], Chap. 3). Having studied graph properties of railway networks and having performed tests, we decided to employ a variant of Moore-Bellman-d'Escopo-Pape algorithm [10]. Our modi cations to this algorithm improved its eÆciency by appropriate pre- processing involving data preparation. Indeed, due to the static nature of the data, a lot of necessary work may be done o -line, so that the data can be transformed to most suitable form before the algorithms are run.

*In nding routes, we adopted no heuristic approach. In this context we* proceed in a similar way as described in [9]. The KRJ engine always tries to

*nd the optimal connection, i.e. the connection with the earliest arrival time* (given the departure time) or, by symmetry, the latest departure time (given the arrival time). The optimality criterion takes into account the weight, or the cost, associated with one change (the p parameter, as described above).

*We di er however from work reported in [9] in the way of creating the* graph on-line and in distinguishing from free arcs and discrete arcs. As a result, the model applied in KRJ search engine is general enough to be ap- plied in di erent, heterogeneous transportation networks, e.g. for road traÆc. In most cases one can assume that the weight of a free arc, determined by the time required to pass the distance corresponding to the arc, is xed. In more complicated situations it may happen that this time (and the weight) will depend on the time of the day, so it will be a function of the start time. Evidently, it is true for travelling on roads (e.g. by cars). Modelling a trans- portation network for cars involves many aspects like topography of streets and actual conditions on the roads. These conditions and, in consequence, the length of travel, may strongly depend the time of the day, the day of the week and on the season (recall the jams on highways on holidays...). Nevertheless, the whole road network may be modelled by free arcs, except for some special

*network segments, like passages by ferries, drawbridges or border crossings* that may by operative only at some times. The latter should be modelled by discrete arcs.

*3 The architecture of Ekspres*

*Moving the o -line version of KRJ to Internet environment required perform-* ing several analysis, design and implementation tasks. Below we present a list of major tasks performed by the team of Ekspres developers.

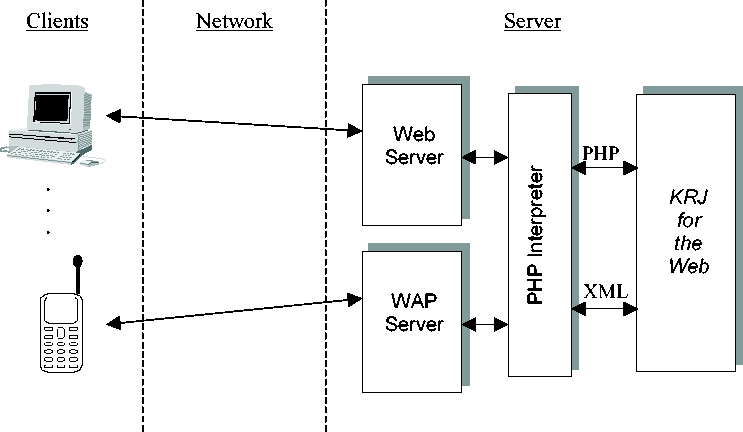
*Design of the architecture of the system, taking into account performance* issues that are of great importance for Web services of such type.

*De ning interfaces between a Web server and KRJ.*

*Making necessary changes to KRJ search engine (changing the interactive* mode to batch mode, changing disk le support, recompiling to a shared library for Linux environment etc.).

*Developing a station names search engine (o -line version of KRJ does not* require such an engine because station names are speci ed interactively as exact strings).

*Fine-tuning the con guration under heavy workload.*

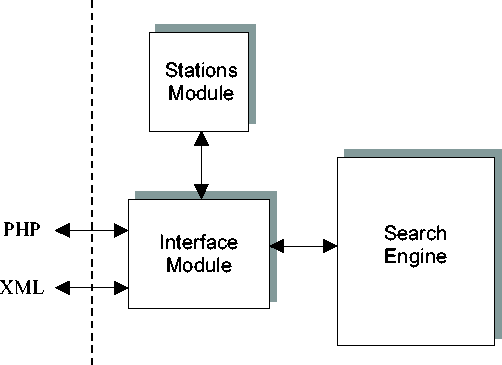


*Fig. 2. The general architecture of Ekspres*

*The resulting architecture of Ekspres is presented in Figure 2. The system* located on a server is composed of four main components: a Web server, a WAP server, a PHP interpreter, and the KRJ for the Web. The system is accessed by its clients via standard Web browsers (or via WAP browsers, if the clients are mobile phones). A user lls in some information in a standard way, into on-line forms that are displayed on Web pages (see Section 5), forming a

*query to the system. The user query is transmitted via Web server and PHP* interpreters to the KRJ for the Web module that responds appropriately. One user interaction may require several queries to be formulated at a client and processed on the Ekspres server, because a user may not be able to formulate a precise query at once or may be interested in di erent connections at di erent levels of detail.

*The KRJ for the Web module consists of several components (see Figure* 3). The main component is Search Engine (described in Section 2) that is responsible for nding connections according to parameters established by Interface Module. Interface Module is also responsible for producing output data for PHP interpreter as set of PHP variables or as a set of XML data. Interface Module co-operates with Stations Module that is responsible for resolving the station names given as parameters of a user query. Stations Module has its own station database independent of the database of Search Engine.



*Fig. 3. Main components of KRJ for the Web*

*The main steps for communicating between PHP interpreter and KRJ for* the Web are given below. The whole communication is performed in a single Linux process created by the PHP interpreter.

*(i) PHP interpreter creates and invokes the KRJ process using parameters* speci ed by a user on the Ekspres main input page (transferred to the PHP interpreter via a browser at the client and the Web server on the server).

*(ii) Interface Module passes names of stations as character strings to Stations* Module and asks the module for identi ers of the stations.

*(iii) In case of exact match, for each station name Stations Module returns* an appropriate station identi er. In case of mismatch, Station Module returns a list of identi ers of those station names that \sound like" the given strings. This list is then transformed into a character string list

*by Interface Module and returned to PHP for presentation to the user in* order to re ne the choice.

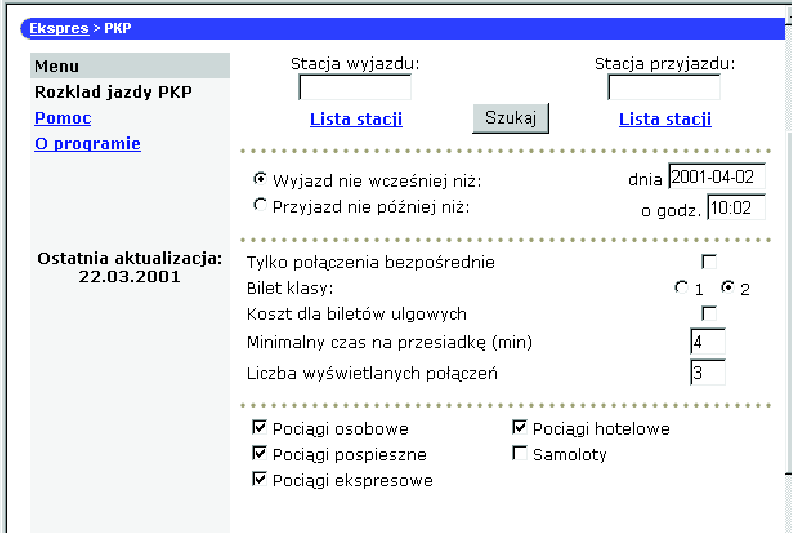
*(iv) The two stations identi ers obtained from Stations Module and other* parameters are passed to Search Engine.

*(v) Search Engine nds connections and returns results to Interface Module.* The results are produced at required level of detail. Interface Module formats the results as a set of character strings (PHP variables or XML text) and passes them to the caller (PHP interpreter that invoked the KRJ process).

*(vi) The KRJ process terminates.*

# *4 User interface*

*The folowing gures present sample of static and dynamic Web pages of the* system. The rst, static page (see Figure 4) is the input page, on which a user formulates a query.



*Fig. 4. The main input page of Ekspres The data to be input by the user include:*

*departure and arrival stations (the two text elds at the top of the page; a* user may select from a list of stations or can enter any string that approxi- mates a station name),

*requested departure or arrival date and time (the two text les below the*

*station names elds),*

*whether changes are allowed (the check box below the date and time),*

*kind of tari to be used for calculating cost of the travel (the next two radio* buttons and one check box),

*how much time (at least) he'd like to have for a change (the next text eld),*

*number of connections to be found (the next text eld),*

*kinds of trains to be used and whether ights are allowed (the series of check* boxes at the bottom of the page).

*The results of a user query are presented in a clear way on dynamic Web pages* on di erent levels of detail. On the rst level of detail the parameters of the user query is displayed, and for each connection found:

*departure time*

*arrival time*

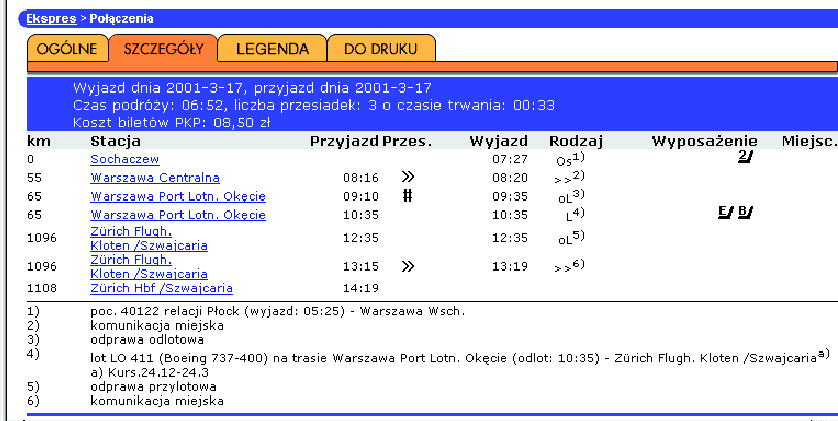
*total time and total length of travel*

*time spent on changes*

*cost of railway tickets*

*transportation means used in the connection.*

*A check box accompanies each text line with data about a connection.* If the checkbox for a given connection is selected, then data about this con- nection will be displayed on the page that presents second level of detail. A sample of such page is presented in Figure 5. This page displays detailed re- sults of a query how to get from Sochaczew (small Polish town near Warszawa, capital of Poland) to Zurich Hauptbanhof on 17th March 2001 in the morning, using all possible means of transportation.



*Fig. 5. A sample of detailed output page of Ekspres*

*The data presented on the page instruct the user that:*

*(i) First, he should take the train at 7:27 to Warszawa Centralna (details of* the train are given in a footnote of the itinerary table, identi ed by an appropriate superscript; the equipment of the train is visualised as icons in the table).

*(ii) Then, he should take a city transport to get to Warszawa Okecie Airport* (distance: 10 kms; approx. duration: 50 mins, from 8:20 till 9:10).

*(iii) The next part of the travel is check-in procedure at the airport (approx.* duration: 1 hr, from 9:35 till 10:35).

*(iv) At 10:35 the ight to Zurich Flughafen Kloten starts (again, details are* in a footnote).

*(v) The ight ends at 12:35.*

*(vi) The next part is check-out at the airport (passport control, collecting* luggage etc.), with approx. duration of 40 mins, from 12:35 till 13:15.

*(vii) Finally, the user should take city transport to get to the destination* (distance: 12 kms; approx. duration: 1 hr).

*Additionally, Ekspres produces some other static and dynamic pages. These* include:

*the page for re ning the station name in case of initial mismatch,*

*a page with descriptive information on stations,*

*a page with explanations of icons and other graphical elements,*

*help pages.*

# *5 Conclusions*

*At the time this paper is written, Ekspres has been operating in working* environment for 9 months. It turned out to be highly eÆcient and reliable system. Its availability is practically 100% high, and response time, under heavy workload, is satisfactory to users. This proves that both the search engine and the architecture of the system have been designed properly. The system is periodically updated, according to changes in the timetable, and the date of last update is explicitly displayed on the main page of Ekspres. It is expected that the system will be further extended to cover these features of o -line KRJ which are still absent in the on-line version (for instance, a map of the travel, presented on demand on di erent levels of detail).

# *Acknowledgements*

*The project of moving KRJ into Internet involved a number of professionals* from university and industry. The major contributors (except of the author) were: unregrettable Janusz Cielatkowski (Technical University of Gdansk),

*Michal Wilde (\MiWil"), and Jacek Kujawa with his team (\Wirtualna Pol-* ska" SA).

# *References*

*[1]* [*http://ekspres.wp.pl*](http://ekspres.wp.pl/) *{ URL for Ekspres site.*

*[2] Deo, N., and C.-Y. Pang, Shortest-path algorithms: taxonomy and annotation, Networks 14 (1984), 275{323.*

*[3] Goczyla, K., and J. Cielatkowski, Finding Optimal Route in a Railway Network, Proceedings of International Conference \Operations Research 1994," Program & Abstracts, Berlin, Aug. 30{Sept. 2, 1994, 115.*

*[4] Goczyla, K., and J. Cielatkowski, Program for Journey Planning in Polish Railway Network, Proceedings of International Conference \Operations Research 1994," Program & Abstracts, Berlin, Aug. 30-Sept. 2 (1994), 269.*

*[5] Goczyla, K., and J. Cielatkowski, Journey Planning in a Public Transportation Network, Proceedings of 17th International Conference on Information Technologies Interfaces ITI'95, Pula (Croatia), June 13-16, 1995, 425-430.*

*[6] Goczyla, K., and J. Cielatkowski, Optimal Routing in a Transportation Network, European Journal of Operational Research 87 No 2 (1995), 214-222.*

*[7] Goczyla, K., and J. Cielatkowski, Integrating Di erent Means of Transport into an On-Line Journey Planning System, Proceedings of 18th International Conference on Information Technologies Interfaces ITI'96, Pula (Croatia), June 18-21, 1996, 365-370.*

*[8] Goczyla, K., and J. Cielatkowski, An Object-Oriented Model of a Heterogeneous Transportation Network for Journey Planning Systems, Proceedings of 8th IFAC/IFIP/IFORS Symposium on Transportation Systems, Chania (Greece), June 16-18, 1997, 265-269.*

*[9] Schulz, F., D. Wagner, and K. Weihe, Dijkstra's Algorithm On-Line: An Empirical Case Study from Public Railroad Transport, Proceedings of 3rd Workshop on Algorithms Engineering, Lecture Notes in Computer Science 1668 (1999), 110-123.*

*[10] Syslo, M. M., N. Deo, and J. S. Kowalik, \Discrete Optimization Algorithms with Pascal Programs," 2nd Ed, Prentice Hall Inc, 1993.*