Automatic Extraction of Business Rules to Improve Quality in Planning and Consolidation in Transport Logistics Based on Multi-Agent Clustering

Igor Minakov [1], George Rzevski [2], Petr Skobelev [1], Simon Volman [1]

[1] The Institute for the Control of Complex Systems, RAS,
Sadovaya Str., 61, 443020, Samara, Russia.
{minakov, skobelev, volman}@magenta-technology.ru
[2] MAGENTA Technology, Gainsborough House, 59 - 60 Thames Street
Windsor, Berkshire, SL4 1TX, UK
{ george.rzevski }@magenta-technology.com

Abstract. The article describes multi-agent engine for data clustering and IF-THEN rules generation and their application to transportation logistics. The developed engine can be used for investigating customer source data, pattern discovery in batch or in real time mode and ongoing forecasting and consolidation of orders and in other cases. Engine basic architecture fits well for both batch and real time clustering. The example of data clustering and generation of IF-THEN rules for one of UK logistics operators is considered. It is shown how the extracted rules were applied to automatic schedule generation and how as a result the quality of schedules was improved. The article also describes an approach, which allows getting orders consolidation from extracted rules. Algorithm of rule search and the obtained results analysis are other points mentioned.

Keywords: multi-agent planning, multi-agent learning, data mining, task and resource allocation, if-then rules, logistics, schedule generation, multi-agent real-time clustering, pattern extraction, orders consolidation.

1 Introduction

Nowadays, transportation scheduling is an urgent and significant task for great number of modern companies that have to operate on unstable and highly dynamic market where demand and supply change rapidly.

The complexity of problem solving results from large quantity and diversity of domain knowledge that should be taken into consideration during decision making process added to absence of problem-solving procedures that represent all peculiarities of contemporary business environment. During schedule generation process, it is necessary to consider individual preferences (including strategies, restrictions, preferences) of all vehicles, drivers, orders and clients. Clear example of such peculiarities is a set of rules used by one of companies that were under consideration:

- Some roads become unavailable for double-deckers after rain, since wet tree branches lower and hinder road traffic.
- Avoid allocating a driver to itinerary that goes near driver's home, since drivers tend to call at their places for a cup of tea.

All these factors make manual re-working an obligatory procedure for schedules created with systems for automatic schedule generation while leaving the re-working process quite a complicated task.

To make full use of problem domain features it is required to formalize all knowledge and criteria that serve as a base for human decision-making in schedule development. However, the implementation difficulties are in the following:

- Sometimes a man fails to formalize rules he or she uses for decision making
- Sometimes a man reports incorrect rules
- If several persons engaged into decision making process, there are rules that do not belong to knowledge of any particular person – these rules belong to a system on the whole

Therefore, to produce high-quality schedules in short terms, the system (or agents of this system – if it is an agent-based system) should automatically detect and take into consideration peculiarities of problem domain. For automatic search for such data dependencies and peculiarities, it is possible to use methods and approaches of Data Mining. Of all classical Data Mining methods, systems for revealing if-then rules give the most comprehensible, the best understood results that are easy to interpret, and if necessary, the results can be manually modified by an expert. That's why in this paper we'll discuss methods of automatic extraction of IF-THEN rules and show how such rules can be applied in real-life tasks.

1.1 Brief overview of algorithms, applicable for if-then rules extraction

Today several software products solve a problem of finding if-then rules on datasets. Mathematical algorithms set an underlying logic for these products, for example, algorithms of bounded combinatorial search (i.e. WizWhy), algorithms of developing decision trees (i.e. See5/C5.0) or algorithms based on association rules extraction (i.e. Apriori, Generalized Rule Induction). The description of these tools is available in [1]. However, classical systems have the following drawbacks:

- These algorithms provide tools to search for if-then rules with THEN part consisting of a single condition. It's a very weak type of rule, as a schedule represents a sophisticated entity made of multiple interdependent fields.
- Algorithms require significant data pre-processing before any analysis can be made, sometimes manually selecting target field, sometimes representing data as a set of zero's and one's etc, which is hard for problem domain experts, as well as tiresome, time consuming and error-prone.
- These algorithms have severe constraints on the number of data fields or unique values for each data field. But usually a schedule is a very complex network with multiple data fields and different values for each field.
- Many of these algorithms have constraints on a very high minimum support level for rule generation, thus missing many important-to-user rules, which have high confidence, but represented in the system by small number of records.

- These algorithms offer tools to search for if-then rules only in a batch mode while present-day requirements for systems of automatic schedule generation demand real time operations to react rapidly on market dynamics .

Hence, it is necessary to develop methods that will allow automatic detection of if-then rules with multiple IF part and will ensure rules' applicability for schedule generation. In addition, these methods should include adaptive mechanisms for extracting and adjusting rules in runtime.

To solve the problem we propose to reject searching and looking through all possible rules, but to find rules that characterize the densest groups of records. The value of discovered rules can be evaluated through calculating probabilistic and representative characteristics for these rules. The proposed approach uses multi-agent clustering to reveal dense groups of records.

To estimate the effectiveness of discovered rules, it is necessary to compare system-generated schedule and schedule produced manually by an operator. Nevertheless, acceptable criteria for metric proximity (similarity) for two schedules resists formalization, therefore such estimations should be executed through expertise.

1.2 Brief overview of clustering algorithms

Clustering is a task aimed at revealing groups of high density in data space. These groups are to be further applied in methods of data analysis, knowledge retrieval or methods for effective information extraction. The following groups of algorithms are available up to the present moment:

- Partitioning algorithm – these algorithms decompose dataset of N objects into K clusters. Examples of such algorithms are: FOREL 2, KRAB, KRAB Heuristic, K-means, K-medoid, CLARANS (Clustering Large Application based on RANdomaized Search)
- Hierarchical decomposition and ordering - these algorithms use iterations to decompose dataset into smaller sets until termination conditions are satisfied. Examples of such algorithms are: FOREL, FOREL OPT, KRAB Heuristic 2, BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies). Decision Trees algorithms can be also placed into this group - C4.5, See5/C5.0, CART (Classification and Regression Trees). Similarly, the following algorithms can be classified into this group: DBSCAN, CURE (Clustering Using Representatives)
- Algorithms which quantize the space - these algorithms quantize the space into finite number of cells and then run all operations on produced cells. Examples of such algorithms are: STING (STatistical INformation Grid-based method), CLIQUE [1, 2, 3].

It is more preferable and more effective to find clusters within sub-spaces of original space for the initial task since multi-dimensional data can include noise or have evenly distributed values for some dimensions (and often there is no way to exclude non-meaningful fields from initial analysis). Among all algorithms mentioned above only CLIQUE can detect clusters on sub-spaces, still restrictions of this algorithm make it unsuitable for the current problem:

- The algorithm operates in a batch mode.

- The algorithm subdivides the space into finite number of cells in advance. This approach is non-applicable in our case, since quality of the obtained results will depend upon initial subdivision of space into cells.
- Found clusters are not organized into hierarchical groups.

To overcome these restrictions, we decided to develop a new clustering algorithm on the base of multi-agent approach [4, 5].

2 Problem Domain Specification (Transport Logistics Domain)

The task concerns developing an optimal schedule to allocate transportation orders (in this case, optimality of a schedule is considered in terms of a specific criterion). An order contains information on cargo properties (such as weight, volume, type, name of client and specific requirement for cargo delivery conditions) and requirements for source location and its time window, destination location and its time window. Orders can be allocated to a limited number of trucks of different types. In some cases, a transportation company can execute an order with 3rd party carriers, however, this leads to profitability decrease. Some orders include additional restrictions on appropriate trucks (for example, chilled cargoes are to be transported on trucks equipped with refrigerating units). Truck drivers also have schedules of their own that regulate working hours, besides driver workday is a subject of safety requirements and labor code. Orders come into system in time moments that are unknown in advance; scheduling is carried out in real time, this means that planning process and execution process go in parallel. Nevertheless, event planning is to be accomplished before event execution commences or before the time when event execution can be considered as delayed. If necessary, event processing breaks off when a new event appears, for example, in case of a truck's breakdown, no more allocations is made to the truck, already allocated orders are re-planned while allocations on other trucks continue to go in parallel.

The key criterion in estimating a schedule is maximization schedule value to an enterprise. The following factors exert influence on enterprise-relevant value of a schedule:

- Cost of schedule (cost of transportation each orders according to schedule)
- Total mileage of all trucks
- Customer satisfaction level of individual clients
- Percentage of successfully allocated orders
- Planning time

More details on the subject are available in [5, 6].

3 Suggested Solution

On the basis of problem domain data a table is created, where each row (record) represents a unique entity of problem domain, for example, a transportation order that includes information on source and destination location of transportation and cargorelated information. Thus, the problem domain is presented as a multi-dimensional space, where each record is an element of this space. This space is heterogeneous, since axes of the given space can be of different types (integer, real, strings, currency, time etc).

3.1 Multi-agent clustering algorithm

Within the approach offered, agents of records and clusters start to play the main role. An agent is a software object capable of analyzing a situation, making decisions, performing actions over objects of a world and of communicating with other agents. Entering clusters, agents form virtual communities similar to temporary hierarchies, which can be organized in different ways.

In the most simple case, records try to find the most "profitable" clusters with the maximum density, which is a metric for its decisions (in more complex cases, metric can include number of records, number of sub-dimensions for cluster, time of life in cluster, type of attributes etc). The process of search is always started with the nearest points and extends gradually. When a record finds a proper cluster, it makes an offer and waits for a reply. The cluster reconsiders its locality, calculates its variant and either accepts or rejects the offer. Thus instead of a centralized optimal "front-office" decision of a classic algorithm, the offered approach provides solutions taken at the lowest level. These solutions are only based on some current local balance of interests of a particular record and some cluster. If each of the sides agrees, the record enters the cluster, if not – the record searches for other variants. The stages of this process in a simple 2D case are shown in Fig. 1, where: a – the first record's entering, b – the second record's entering and creating of the cluster, c – the third record's entering, forming the second cluster of the first cluster and the third record, d – the second cluster "lures" the records from the first cluster to enter it (as it's two-dimensional and based on selected value formula is more "profitable" for records), the first cluster is dropped out, the fourth record comes, e – a new cluster is formed, f – thea new cluster "lures" the records from the inner cluster, a new record comes, g – a new record comes and the process starts from the very beginning, h – the final cluster.

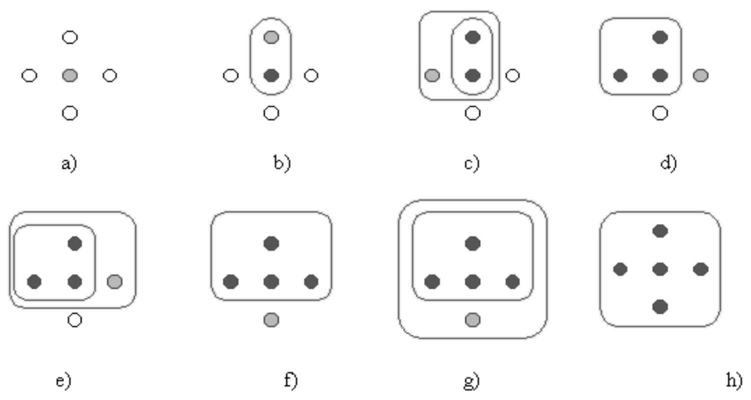

Fig. 1. The stages of forming a cluster within the approach developed.

In more complicated cases, agents can interact in the virtual market and use money, which regulate the records' abilities or restrict them in making decisions in a natural way. For example, in transportation logistics, the sum of money available for a record can be set as a price for order, which customer is due to pay. Then the record of VIP order is "richer" than a record of a single order from random customer. This makes it possible for the first record to form clusters with distant records, taking into account bigger context, and generating more clusters as a result. In this situation different models of the microeconomics of cluster or records can be applied. For example, clusters can take charges from records for the right to enter them in accordance with the "club system" model (the amount of payment does not depend on the situation, richness of a record or the number of the club members) or according to the "shareholder" model (the amount depends on the situation). In the latter case, a record can even increase its capital entering or quitting the cluster in the right time. It is obvious that if a record sees a more beneficial cluster and it does not have enough money to enter it, it can leave the current cluster, which may cause a wave of the process of dropping out cluster. In the process of interaction of clusters and records real negotiation is conducted, with parties meeting each other half-way, for example, by mutual concessions. All this provides different variants of taking approximate decisions and the control of the correlation between the accuracy and cost and time effectiveness.

A case is even more complicated in the situation with time dependencies, when the processes of self-organisation can be supplemented by the processes of evolution. In this case, clusters and records pay time taxes for acting in the system and for staying in clusters. This will have an effect on their financial resources, making some clusters and records disappear from the system, decreasing thus the load on the system, while some of them will be growing and becoming stronger. In this case, clusters and records are supposed to think not only about "money", but about their "lifespan", balancing between the criteria. For example, if a cluster is under threat to be dropped out of the system, when it does not receive any offers from records for a long time and thus is not paid the entrance fee, it can considerably decrease this fee to attract new members. This will help the cluster to stay in the system for some time.

These examples make it clear that clusters and records can make decisions according to different rules, combining, for example, a simple local density and "public value" (the number of clusters a record participates in or the number of records in a cluster), "richness" or 'lifespan" of a cluster or a record. These combinations of rules depend on what a user wants to see: bigger and more steady groups of clusters or, on the contrary, the most dynamic and miniature; densest or the most disperse; "the richest", the most "die-hard" or any others.

The clustering processes for a stable data structure resembles crystallization processes – records create various structures at the micro level, the produced structures on their part start to participate in clustering process. The process stops when the whole structure is crystallized. The outcome of the process generates high-level structures, which are more or less stable, but are adjusted in real time as new records (events) come into system. Further information on clustering algorithm is available in [4, 7].

3.2 If-Then rules extraction algorithm

Further, it is necessary to transform clusters into a form that can be easily understood by a problem domain expert or by an automatic system, in other words – rules. To produce a rule on the base of cluster, all axes of the space are subdivided into two groups regarding particular features of problem domain – first group represent fields, which can be part of IF part of rule, and second – of THEN part (if such groups are not specified, it is considered as each field can be either in IF or THEN part of different rules). Next, retrieval of data for various dependencies starts – how fields of one category depend upon fields of another category. The guideline for space subdivision is quite simple – one group includes fields over which we have a control and which we can manipulate (information is being defined at schedule generation stage) while the other group includes fields unavailable for control (order information). In analysis we neglect clusters made of fields only from one category.

Let us consider cluster as a logical rule of the following sort IF (condition A_1) and (condition A_2) and … (condition A_N), THEN (condition B_1) and (condition B_2) and … (condition B_M), where A – conditions relevant to axes from the second category (fields over which we have no control), while B_i – conditions relevant to axes of the first category (fields over which we have a control). For example, if an order requires transportation of a cargo of 5 kilograms, then this order should be allocated to the truck of "Gazel" type that belongs to Trans-GAZ carrier-company.

We can distinguish three main characteristics of each rule that help to evaluate the rules produced:

- Representativeness – shows number of elements that correspond to the given rule (for IF part).
- Confidence level – shows how many elements of domain space that correspond to the left part of the rule (IF part) meet conditions of the right part of the rule (THEN part). This parameter depends upon applied pattern (for example, among all ones that are parous, all are women; while not all women are parous).
- Completeness – shows how many elements of space meet conditions of the right part of the rule (THEN part) and do not meet left part of the rule (IF part). For

example, “if you are a human being, then you are mortal” rule has high level of representativeness but a low level of completeness (since not all mortal beings are human beings).

The higher value of representativeness and confidence level has a cluster, the more valuable the revealed interdependency is for an expert (with the exception of obvious rule or tautology).

When the rule is detected, we can try to move some conditions from the reason part of the rule (IF) into the consequence part of the rule (THEN). If this operation does not decrease a confidence level of the rule, then the modified rule is more preferable as it contains fewer conditions in IF part. For example, the following rule “If an order requires transportation from source location Krasnoyarsk then destination location of an order is Moscow and this order should be planned on truck of ZIL type” is more preferable than the rule “If an order requires transportation from Krasnoyarsk to Moscow, then this order should be planned to truck of ZIL type” (if confidence level after transformation does not decrease).

Note that any rule formalized in terms of multi-dimensional space will represent a certain group of elements of high density, in other words, a cluster. Hence, any rule with high level of confidence represents a cluster (inverse statement is not true in general case); therefore, if clustering algorithm finds all dense groups, it finds all rules with high level of confidence, so it’s reasonable to use clustering algorithm to extract IF-THEN rules.

4 Integration with Logistics i-Scheduler

A developed product for extraction rules in logistics domain (it was named Pattern Seeker) is also able to integrate with logistic schedulers, thus monitoring a planning application in real-time mode and providing rules, which can be applied to improve the quality of planning in a constantly changing dynamic environment. A possible scheme of integration with operational planner Magenta Operational i-Scheduler is given below (Figure 2). Magenta Operational i-Scheduler is an agent-based Intelligent Scheduler for Road Transportation real-time Applications [6].

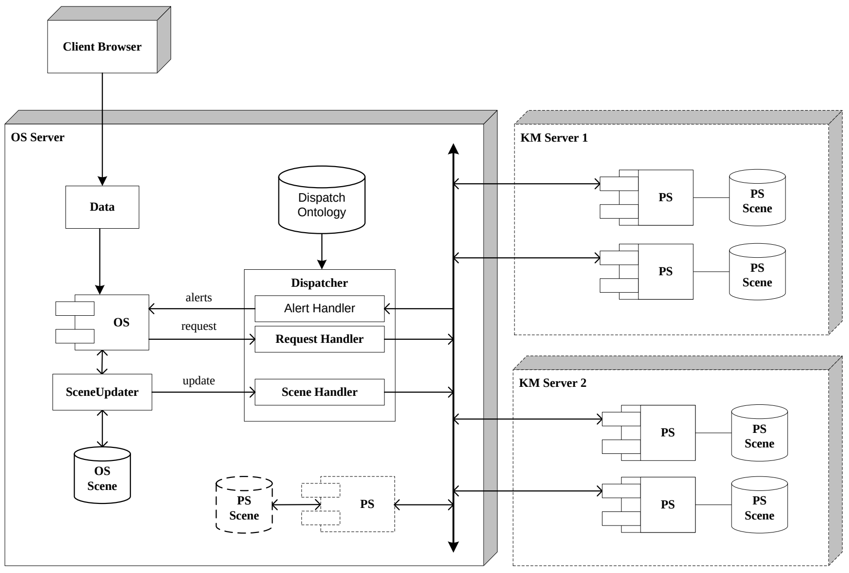

Fig. 2. Integration of Pattern Seeker with Magenta Logistics i-Scheduler. (OS – Magenta Operational i-Scheduler; PS – Pattern Seeker; Dispatcher – main module, which is responsible for parallel work of several applications; OS Scene – ontological scene of OS, which represent a transportation network; PS Scene – ontological scene, which is given as an input to PS)

Pattern Seeker is located on a separate computer or several computers (as its algorithm is parallel). In some cases, it can exist on one computer together with OS, but this variant can lead to OS performance decrease, which will require considerable calculation resources.

The two systems interact through dispatcher module. Both modules interact with dispatcher module by means of asynchronous messages.

Dispatcher availability gives the following advantages:

- An ability to add new Pattern Seeker instance.
- An ability to change any Pattern Seeker instance location.
- Modules interaction transparency and unification gives an ability to integrate the system with modules of other producers.

OS and Pattern Seeker work independently

When the two systems work together, the exchange of the following messages takes place:

OS sends Pattern Seeker the following messages:

- Notification about new order that has come into the system
- Notification about schedule changes
- Query about some certain order planning limitations
- Query about preferred consolidation for non-allocated order

Pattern Seeker can send OS the following messages for agents:

- Proposals on joining of non-allocated orders into one consolidation
- Proposals on adding new orders to a trip for backhaul

- Advice to re-plan some certain orders

This process provides a learning basis for OS agents, as each instance of each agent type applies its strategies for selecting best route or best order or consolidation option, and depending on recommendations selects best appropriate in the current market conditions, changing set and/or priorities of strategies.

5 Data of logistics company focused on Transportation Scheduling

During the cooperation with the logistics company a problem was set to automatically generate a schedule that would be similar to a schedule developed by an operator manually. However, the customer failed to define metric criteria to estimate proximity and similarity of schedules, thus, schedule estimation was run through expert evaluation.

The following points specify project purpose in more details:

- Plan customer orders and provide schedule using own and 3rd party fleets
- Ensure that the generated schedule is sufficiently workable from cost perspective.
- Consider all feasibility constraints and preferences, which are now possessed by operators (patterns of delivery, preferred carriers, trip shape)
- Consider bringing future orders to optimize today's trips
- Validate the use of human-like heuristics in a real environment (allocation of constrained orders/resources first, working from most distant/close points, etc)

The customer provided a dataset made of 920 orders planned manually by operators. To enable the system of automatic schedule generation produce a schedule similar to the one created by a human operator, we considered a task of extracting hidden rules from the test dataset that served as a base for a schedule development. Further information on the system for automatic schedule generation is available in [6].

A table with information on orders and the way how orders were planned was created where each row represents an order. The following fields were included into the table (each field represents space axis):

- From – the source location of cargo transportation;
- To – the destination location of cargo transportation;
- Pallets – number of units to be transported;
- CarrierType – name of carrier company that was selected for order execution;
- Depot – location of vehicle that should execute transportation;
- VehicleType – type of vehicle that should execute transportation;
- Orders – total amount of orders transported in parallel on a vehicle;

Based upon characteristics of the problem domain, we selected the following pattern: Form, To, and Pallets axes are set as axes of the second category (we do not have control over these fields), while CarrierType, Depot, VehicleType, and Orders axes are set as axes from the first category (we do have control over these fields).

The developed algorithm found 218 rules. Figure 3 shows how found rules are distributed by confidence level.

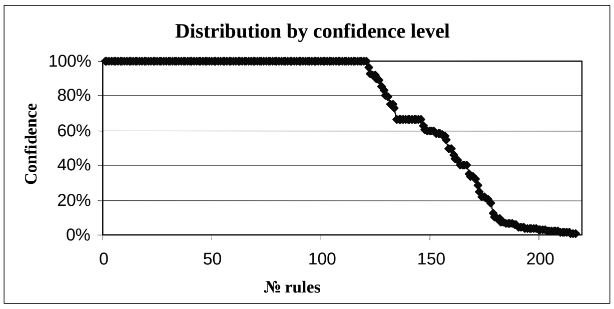

Fig. 3. Distribution of System-detected Rules by Confidence Level.

As the graph shows, more than half of rules have confidence level of 100%, these figures ensure high value of the detected rules.

Table 1 shows an example of system-found rules, companies and geographical locations which were renamed by customer demand.

Table 1. Example of the Rules Detected by the System.

№	Pattern	%	cou
182	<To> = "Bryansk" -> <From> = "Moskow" ,<Carriertype> = "Bryansk TRANS" ,<Depot> = "Bryansk" ,<Pallets> = "26" ,<Vehicletype> = "ZIL"	100	37
110	<To> = "Samara" -> <Carriertype> = "Moskow" ,<Depot> = "Moskow" ,<Vehicletype> = "ZIL" ,<Orders> = "11"	100	33
52	<To> = "Archangelsk 1" -> <From> = "Volgograd" ,<Carriertype> = "TRANS GAZ" ,<Depot> = "Archangelsk" ,<Pallets> = "26" ,<Vehicletype> = "Gazel" ,<Orders> = "3"	100	30
111	<From> = "Archangelsk 3" ,<To> = "Kiev" -> <Carriertype> = "TRANS GAZ" ,<Depot> = "Kiev" ,<Pallets> = "26" ,<Vehicletype> = "Kamaz" ,<Orders> = "3"	100	30
8	<To> = "Bryansk" ,<Pallets> = "26" -> <From> = "Kiev" ,<Carriertype> = "TRANS GAZ" ,<Depot> = "Kiev"	100	28
101	<From> = "Samara" ,<To> = "Volgograd" -> <Carriertype> = "TRANS GAZ" ,<Depot> = "Volgograd" ,<Vehicletype> = "Gazel" ,<Orders> = "11"	100	28
132	<To> = "Bryansk" ,<Pallets> in [1.0 .. 3.0] -> <Carriertype> = "Bryansk TRANS" ,<Depot> = "Bryansk" ,<Vehicletype> = "Kamaz" ,<Orders> = "11"	100	26
143	<From> = "Krasnoyarsk" ,<To> = "Moskow" ,<Pallets> in [1.0 .. 3.0] -> <Carriertype> = "Bryansk TRANS" ,<Depot> = "Bryansk" ,<Vehicletype> = "ZIL" ,<Orders> = "11"	100	24

*Names of geographical locations are obfuscated to keep confidentiality of client's data

System-detected rules were demonstrated to an expert who confirmed most of the rules and proved the revealed dependencies to be intrinsic characteristics of the problem domain. Even more, experts did not report in advance approx. 8-12% of the rules, which were found by the system, although these rules have great value of confidence level.

All obtained rules were loaded into the knowledge base of the system for automatic schedule generation.

Schedule calculation time:

- Without Pattern Seeker patterns, for 1 week period - ~ 5 hours
- With usage of Pattern Seeker patterns, for 1 week period - ~3 hours

Moreover, expert evaluations for the following were performed:

- Analysis of schedule created by experienced human-operator;
- Analysis of system-generated schedule (without use of extracted rules);

- Analysis of system-generated schedule with extracted rules applied;

Schedules were developed on another dataset of the customer company that was different from one served as base for rule detection. The expert declared the system-generated schedule where revealed rules were applied, as the one that was the most similar to the schedule created by experienced human-operator.

Implementation of the algorithm brought gains in schedule quality. The following figures show quality grows after application of Pattern Seeker-found rules:

- Manual rework needed – decreased by 32%
- Trip quality – increased by 17%
- Gaps presence in the trips – decreased by 11%
- Fleet mileage – decreased by 16%
- Fleet usage – decreased by 8%

In overall, it brought approx. 20% of increase in schedule quality (To obtain more information of how schedule quality is calculated, and by which parameters it can be affected, please refer to [6]).

The application of this system will considerably decrease the time required to customize logistic i-scheduler to customer problem domain. According to estimations, the customization period will drastically decline from 1-2 months to 10-15 days.

6 Consolidation example

Finding possible options for consolidation within transport logistics is one of the specific cases for applying clustering algorithm and running clustering analysis. The clustering algorithm is applied by setting specific filters on IF and THEN parts of the rules. This is useful when the rules consolidating fields for geographical coordinates, time windows and Journey-Time Matrix information are of particular interest. In that case, each cluster can be considered as a group of orders that are potentially suitable for consolidation.

The following parameters should be considered:

- Nearness of source locations by geography and by distance/time (JTM)
- Nearness of destination locations by geography and by distance/time (JTM)
- Fully adequate intersection of time intervals so that if one truck takes all orders of consolidation, it manages to deliver them in time
- Different sizes of trucks that can take the group

At the next step, the heuristics are applied when the consolidation groups are found. It is necessary to define for each cluster the way in which consolidation should be treated within this group – whether all orders in the group are to be shipped by one truck or orders in the group are to be shipped by several trucks that are similar in characteristics. The decision depends upon distance between locations, since time is required to load each cargo and to deliver each cargo, whilst driver's working hours are bounded. In addition, the decision depends upon cargo properties, in some cases, a special-purpose truck is required to deliver a cargo, for example, for chilled cargoes.

Additional clustering run on the consolidation clusters helps to bring to light more dependencies and details. When tested on real data the system demonstrated the following characteristics:

- 90% of orders found consolidations with at least 1 order
- 423 good consolidations (20-26 pallets) were found

Examples of consolidations found can be seen in Figure 4.

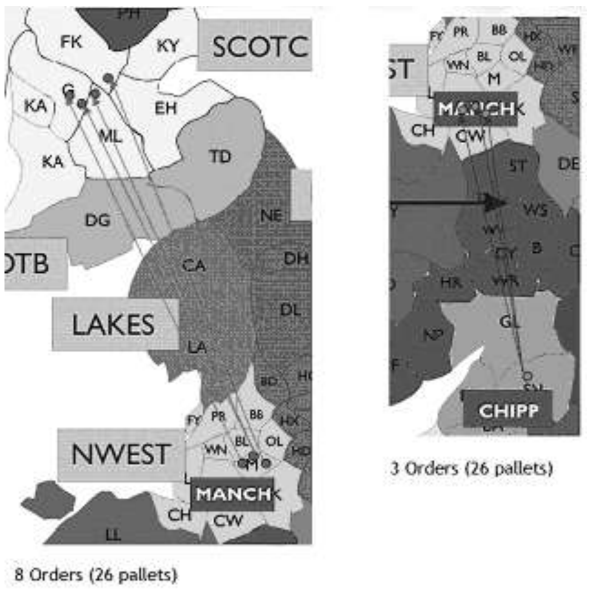

Fig. 4. Example of Consolidations Found.

More detailed results are given in Table 2

Table 2. Results of Consolidations Found.

	Orders			
Pallets	Natural Consolidation	Inbound	Outbound	Pattern Seeker
26	715	736	1202	1250
25	61	52	96	79
24	35	63	71	77
23	83	77	48	44
22	66	66	71	67
21	74	57	60	50
20	60	51	24	30
Total >= 20	1094	1102	1572	1597

As a result with a help of Pattern Seeker it was possible to find more than 45% of consolidations, resulting in improvement of generated schedule quality by more than 15%.

7 Conclusion

The following advantages for client use were achieved through applying offered approach to various real-world logistics applications from different problem domains:

- Significant improvement in quality of automatically generated schedules due to the usage of extracted business patterns;
- Considerable assistance for operators in developing schedules in manual mode or through semi-automatic mode that increases efficiency and accuracy of decision making;
- Ability to validate existing human heuristics and find unknown ones;
- Enhanced knowledge on problem domain that leads to great improvements of scheduling model;
- Decreasing time required to customize scheduler for client problem domain
- Revealing interdependencies hidden in problem domain and capability to spot in time changes in existing dependencies to affect quality of integrated scheduler.

The current approach can be further used for dynamic validation or adjustment of patterns and to improve agents planning strategies according to new incoming events. The approach also applies to a preliminary investigation of problem domain to find more dependencies in data through data analysis and rule retrieval. The produced results can be further used for schedule development either manually or with systems for automatic schedule generation.

Furthermore, the presented investigation showed the following scientific results:

- The new algorithm of searching for if-then rules with multiple THEN part was developed and implemented.
- Revealed rules were applied to the system for automatic schedule generation. Problem domain experts assured that the schedules generated through automatic systems where these rules were applied proved to be similar to the manually created schedules.
- At the same time, system-generated schedules produced on the base of rules exceeded in quality these system-generated schedules where no rules were applied.
- A framework was developed as a base for a new generation of software products that adapt, learn and evolve over their life cycle.

References

1. Larose D.: Discovering Knowledge in data. Wiley Interscience (2005)
2. Agrawal R., Gehrke J., Gunopulos D., Raghavan P.: Automatic subspace clustering of high dimensional data for data mining applications. Proceeding of the ACM SIGMOD Conference on Management of Data, June 2-4, Seattle, Washington, USA, Publisher of proceedings (1998) 94-105.
3. Ester M., Kriegel H., Sander J., Xu X.: A Density-Based Algorithm for Discovering Clustering in Large Spatial Database with Noise. Proceeding of 2nd International Conference on KDD, August 2-4, Portland, Oregon, USA, Publisher of proceedings (1996) 226-231
4. Minakov I., Rzevski G.: Skobelev P.: Data Mining. Patent Reference No: GB 2 411 015 A. Published 17.08.2005.

5. Rzevski G., Skobelev P.: Agent Method and Computer System For Negotiating in a Virtual Environment, Patent Reference No: WO 03/067432 A1. Published 14.08.2003.
6. Himoff J., Rzevski G., Skobelev P.: Multi-Agent Logistics i-Scheduler for Road Transportation: Proceedings of the AAMAS'06, May 8-12, Hakodate, Hokkaido, Japan, Publisher of proceedings (2006) 1514 – 1521
7. Andreev V., Batishchev S., Ivkushkin K., Minakov I., Rzevski G., Safronov A., Skobelev P.: MagentA Multi-Agent Engines for Decision Making Support // International Conference on Advanced Infrastructure for Electronic Business, Science, Education and Medicine on the Internet (ISBN 88-85280-63-3), L'Aquila, Italy (2002) 64-76

Published: Minakov, I., Rzevski, G., Skobelev, P., Volman, S. Automatic extraction of business rules to improve quality in planning and consolidation in transport logistics based on multi-agent clustering // 2nd International Workshop Autonomous Intelligent Systems: Agents and Data Mining, AIS-ADM 2007; St. Petersburg; Russian Federation; 3 June 2007 through 5 June 2007. Volume 4476 LNAI, 2007, Pages 124-137.