Reference: 42271
Issue 1 (10/12/01)

iOpt toolkit

An optimisation solution for e-business

Contents

Offices and laboratories worldwide

Adastral Park, Martlesham
Ipswich IP5 3RE, UK

Email
btexact@bt.com

*Free*phone
0800 169 1689 (UK only)

Phone
+44 (0)1473 607080

Fax
+44 (0)1473 607700

Published by BTexact Technologies -
a division of British Telecommunications plc

Registered office: 81 Newgate Street, London EC1A 7AJ.
Registered in England, number 1800000.

iOpt toolkit - an optimisation solution for e-business

The iOpt toolkit is designed to support rapid development of e-business applications that require planning and scheduling functions. Written entirely in the industry-standard Java programming language, it uses advanced artificial intelligence techniques based on heuristic search.

State-of-the-art problem modelling

The iOpt toolkit incorporates cutting-edge technology to facilitate problem modelling, offering declarative programming capabilities within Java.

Part of the toolkit is the Invariant library. The Invariant library provides built-in data types such as integer, real, boolean, string, object and also set versions of these types. Arithmetic, logical, set and other operators are available allowing the user to state its problem (i.e., decision variables, parameters, constraints and objective function) in a form that is close to the mathematical formulation.

Invariants are basically lightweight constraints performing one-way propagation. They can be configured to form arbitrary networks to model even the most complex of problems. They are particularly suited for heuristic search techniques and interactive visualisation since both these areas require an efficient way to assess the impact of changes on solutions of optimisation problems.

Solution changes (also called moves) are the foundation for heuristic search methods and they are iteratively used to improve a starting solution for the problem. Devising an efficient move evaluation mechanism normally requires a person with significant expertise in the area. This hinders the widespread use of heuristic search. The Invariant library addresses this problem by providing generic algorithms which can achieve efficient move evaluations in an automated way without requiring any particular expertise from the user. The Invariant library is generic and its applications can be as diverse as computer graphics and software modelling.

The problem modelling framework uses the Invariant library to offer modelling facilities particularly customised for heuristic search. It includes all foundation classes for problem modelling. In addition, it offers solution management facilities which can be exploited by algorithms which work on a single solution (e.g., local search) or a population of solutions (e.g., evolutionary computing). The framework is the foundation for more domain-specific frameworks such as the scheduling framework.

Ready to use models for scheduling problems

Scheduling problems can be found in diverse sections of the economy (e.g., manufacturing, transportation/logistics, utilities etc.). To assist non-expert users to develop applications in these areas, we implemented a framework of Java classes for specifically modelling scheduling problems. These classes hide the complexity of the decision model from the user who focuses on problem modelling with entities from his application domain (e.g., activity, resource, break etc.).

The scheduling framework is rich in the facilities offered. There is a class hierarchy for activities to capture the different types found in applications. Depending on the type of the activity, the user can state various resource and time constraints (e.g., task A before/after task B, time windows, etc.). For the resources, the framework offers different types of timelines such as state, capacity and capability timelines. The interaction of activities with the resource timelines can capture all the necessary constraints related to the execution of activities on resources. The scheduling framework can also model resource setup/travel times. These are sub-models which can be easily implemented by external systems, such as geographic information systems in the case of travel times, to offer realistic estimates. The cost objectives considered are generic and can be configured to include as many terms as required to capture the particular business problem. Standard built-in costs include unallocated costs for activities, resource usage costs, time window/lateness penalties, overtime, travel and service time costs as well as resource-activity compatibility preferences.

The scheduling framework internally uses the problem modelling framework. This makes it fully capable to accommodate the different variations of scheduling problems in terms of additional decision variables, constraints or costs that are not already provided. In the toolkit, we include examples of how the framework can be used to model problems from three areas:

- Transportation
- Manufacturing
- Workforce scheduling

Heuristic search methods for optimisation

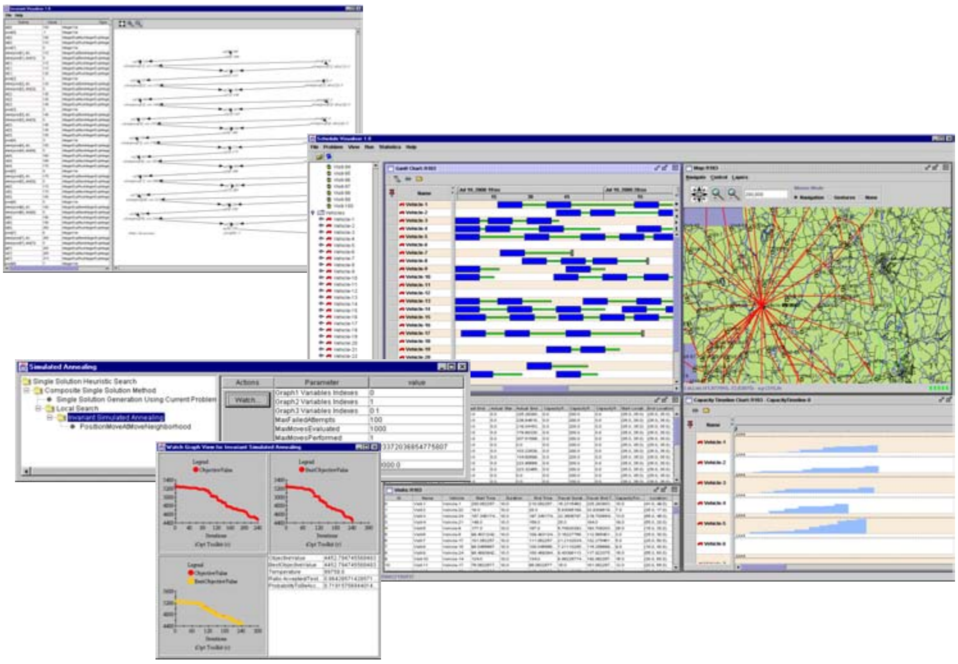
Heuristic search is a very popular class of advanced optimisation techniques. The class includes algorithms such as local search, simulated annealing, tabu search, guided local search, and genetic algorithms. These methods are known for their flexibility in terms of modelling problems but above all for their performance since they have been used to find the best known solutions for many challenging benchmark instances of combinatorial optimisation problems. The iOpt toolkit provides the heuristic search framework which allows the user to build heuristic search methods such as those mentioned above. There is a deliberate separation between the problem modelling framework and the heuristic search framework in iOpt. This separation allows the user to run many different algorithms on the same model choosing the best one for his/her application.

The heuristic search framework can support a wide range of heuristic search methods. To achieve this flexibility, we went through the process of classifying the many heuristic methods available in the literature and identifying patterns that are common to these methods. These patterns then formed the core of the problem solving framework while the points of variation between different algorithms (also known as hotspots in software engineering) became explicit, allowing a great number of algorithmic variants to be implemented and incorporated to the rest of the architecture with minimum effort. The final product is a flexible set of classes for neighbourhood search, move operators, meta-heuristics, single solution and population solution techniques, implementing the various building blocks required to compose a heuristic search method. The user may use these blocks, extend them or provide their own versions. In addition, the toolkit offers event-based heuristic search where the user can capture events from the search process to define its own search strategy. In this way, a great number of algorithms can be synthesised using the toolkit. The toolkit can be extended to incorporate new methods and/or used to build novel methods.

Interactive visualisation

For real world optimisation systems to be useful, they require not only sound algorithmic frameworks but also intuitive visualisation facilities to enable the user to participate in the problem solving process. The iOpt toolkit incorporates ready-to-use visualisation components* which can be plugged in to the different algorithmic frameworks included in the toolkit. This facilitates the development of interactive optimisation systems that allow what-if scenario testing along with intelligent manual interventions. All the visualisation components are based on Java Swing technology and can be easily integrated in any customised graphical user interface (stand-alone or web based).

* These components are built on top of third party software which has its own licensing agreement



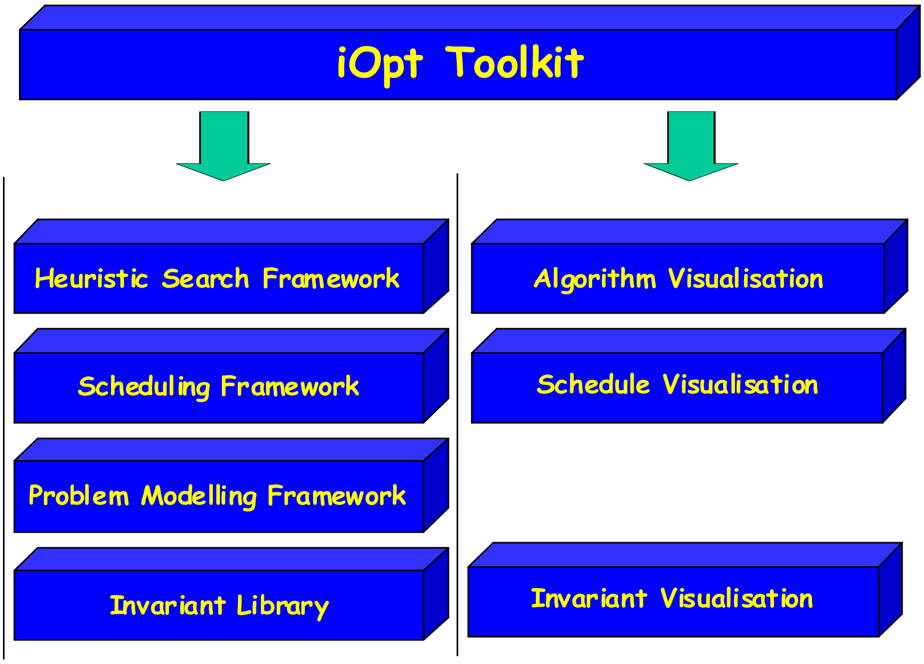
- **Invariant visualisation:** This package offers ready-to-use components to visualise an Invariant Network. The MVC approach is followed and the visual components use the observer-observable pattern (which is fully supported by the invariant library) to connect to the constraint network. The user may modify or read the values of variables in a table view or use a graph* view to visualise the structure of the constraint network. The components in this package can be incorporated into customised applications.

- **Schedule visualisation:** This package offers ready-to-use components to visualise problem models based on the scheduling framework. As with invariant visualisation, the model view controller approach is followed. A number of interactive commonly-used components are included such as schedule tree navigator, gantt chart*, capacity timeline chart*, map*, objective function view, along with table views for resources, activities, etc. Since these components are directly connected to a schedule model using the MVC architecture, the user may easily drag and drop activities or change parameters and immediately see the impact they have on the schedule constraints and objectives.

- **Algorithm visualisation:** Using special graphic components the user can examine the structure of algorithms view/change their parameters and monitor their performance.

* These components are built on top of third party software which has its own licensing agreement

iOpt toolkit and e-business applications



The iOpt toolkit offers a full range of components for building optimisation systems in the Java programming language making it ideal for e-business applications. Because the toolkit is based on heuristic search, it offers superior flexibility in problem modelling combined with advanced optimisation algorithms which can deal effectively and efficiently with even the hardest of combinatorial optimisation problems. This is in contrast to existing optimisation libraries and systems that are based on C++ and techniques such as mathematical programming or constraint programming, which are known only to either be good optimisers or offer modelling flexibility, but not both. This is especially the case when it comes down to solving combinatorial problems. This is due to the nature of their methodology which follows the enumerative approach rather than the heuristic approach used in the iOpt toolkit.

Because it is written entirely in Java, iOpt combines easily with the latest technologies in software development such as enterprise Java Beans, Java servers/servlets, web-based/Java Swing-based graphical user interface, etc. iOpt can be used in the new generation of e-business applications to optimise the use of resources and maximise customer satisfaction. Application areas include field resource planning and scheduling, call centre resource management, enterprise resource planning, manufacturing, distribution, logistics, collaborative planning and scheduling in business to business, advanced planning and scheduling, application service platforms, product configuration and others.

Furthermore, iOpt can help the telecommunications industry to optimise the development and operation of the new Internet infrastructure. Applications can be developed using iOpt for broadband/UMTS network planning, design and routing tasks. Ongoing research and development aims to extend the toolkit with exact search methods, heavyweight constraints, algorithm visualisation components and dynamic heuristic search algorithms. This is in parallel to code improvements, testing and enhancement of the existing components.

Specifications

Invariant library

This library facilitates problem modelling. Advanced propagation algorithms offer automated/efficient move evaluations.

- Declarative programming in Java.
- One-way functional constraints with lightweight implementation.
- Rich set of variable types and operators (e.g. integer, real, boolean, string, object).
- Two-phase propagation algorithm supporting lazy/eager evaluations.
- Incremental update for aggregate invariants (e.g. sum).
- Constraint propagation priorities.
- Dynamic invariants (e.g. If-then-else, array indexing, object fields).
- Set operations and queries (e.g. Union, Intersection, Select, Reject).
- Generic so that it can be used elsewhere (e.g. Computer Graphics, UML/Software Generation).
- Transaction interface with roll back facilities for changing the values of variables.

Problem modelling framework

This framework offers classes for all the basic entities of a problem model.

- Domain variables of different types.
- Base classes for problem and solution entities.
- Automated management of best/current solution, populations of solutions.
- Ability to stop move propagation when infeasibilities are detected at an early stage.
- Users can add/remove objectives, constraints and decision variables.
- Basis for specific domain frameworks (e.g. scheduling framework).

Scheduling framework

Ready-to-use classes for modelling planning/scheduling problems.

- Based on problem modelling framework.
- Activity modelling facilities (tasks, breaks, overtime, start/end resource activities).
- State, capacity, capability timelines.
- Activities interact with timelines, resources impose constraints on their timelines.
- Task time windows, precedence relations, resource relations.
- Built-in costs for unallocated tasks, delayed tasks, various resource usage types (e.g. fixed costs, overtime, service, setup/travel).
- The user can extend the build-in costs or define his/her own costs/objectives.
- User-defined service model covers (resource/task compatibility, fixed/time related service costs, service duration).
- User-defined setup/travel model covers (resource/task compatibility, fixed/time related setup/travel costs, setup/travel duration).
- Allows for integration with duration estimation modules for service, setup, or travel (e.g. based on GIS, historical records, etc.).

Heuristic search framework

The framework offers extensive facilities for building heuristic search algorithms.

- Covers single-solution (e.g. local search) and population-based methods (e.g. genetic algorithms).
- Includes solution generation and improvement classes.
- Built-in commonly-used solution representations.
- Built-in move operators, neighbourhoods which work on these solution representations.
- More than15 ready-to-use neighbourhood search methods (e.g. First improvement, best

improvement, aspiration plus, random, threshold based).
- Composite neighbourhoods and strategies for implementing variable neighbourhood searches.
- Built-in single solution meta-heuristics including simulated annealing, tabu search, and guided local search.
- Search process events can be dispatched to listeners. This event mechanism can be used to build novel meta-heuristics.
- Templates for implementing perturbation/restart strategies.
- Population-based meta-heuristics such as genetic algorithms including support for mutation and crossover operators as well as building hybrid methods using local search as the mutation operator.
- Facilities for incorporating additional population based methods (e.g. ants, scatter search, adaptive memory programming).
- Controls for configuring the various components, stopping the search, collecting search statistics.
- Mechanisms for implementing hybrid methods which use a number of generation/improvement strategies in a sequence.
- Ability to declaratively represent algorithms using the invariant library.
- Can be used for both practical applications and also research and development in new algorithms.

Visualisation

This package includes a number of plug and play components which can be used in conjunction with the optimisation software. The components follow the MVC architecture and they are swing-based.

- Constraint graph component.
- Constraint table component.
- Schedule tree navigator component.
- Schedule gantt view component (allows interactive task scheduling).
- Schedule map view component (allows interactive vehicle/technician routing).
- Schedule timeline view component for visualisation of resource capacities.
- Ready to use table views for aspects of the scheduling model (tasks, breaks, resources, schedules, etc.).
- Graphical wrappers for invariants, schedule cost function, etc.
- Algorithm tree view component.
- Algorithm parameters table view component.
- Various graphs for monitoring algorithm parameters over time.

BTexact Technologies **www.btexact.com** Adastral Park, Martlesham, Ipswich, Suffolk IP5 3RE, UK
email: btexact@bt.com phone: **+44(0) 1473 607080** fax: **+44(0) 1473 607700**