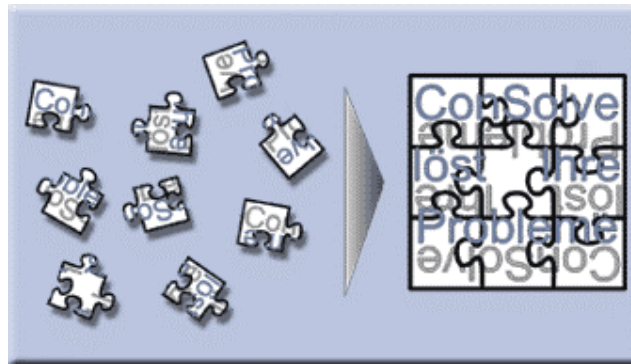


Optimierungskomponente ConSolve®

In vielen Anwendungsgebieten für Software sind Optimierungsprobleme zu lösen. **ConSolve®** gehört zu einer neuen Generation von constraint-basierten **Optimierungskomponenten**.

Die abstrakte Beschreibungssprache von ConSolve erleichtert es dem Anwendungsentwickler Optimierungsprobleme zu formulieren, ohne über besondere Kenntnisse auf diesem Gebiet zu verfügen. ConSolve besitzt außerdem eine hochwertige Programmierschnittstelle, so dass die Optimierungskomponente mühelos in Anwendungssoftware integriert werden kann. Durch die moderaten Kosten dieser Technologie wird es nun möglich, hochwertige Optimierungstechnik in Standardsoftware (CTOS) zu integrieren.



Einsatzgebiete

Unsere Optimierungskomponente löst für Sie Probleme aus den Gebieten Schicht- und Stundenplanung, Ablaufplanung, Konfiguration, Produktions- und Ressourcenplanung. Entwickeln Sie eine Anwendung, in der logische Bedingungen eine Rolle spielen, so ist unsere Komponente das ideale Werkzeug für Sie.

Ein gutes Anwendungsbeispiel für ConSolve® ist unsere Dienstplan-Software: Hier wird das schwierige Problem der Dienstplanung (np-schwierig) durch den Optimierer automatisch gelöst.

Architektur

Die Architektur von ConSolve® basiert auf der Trennung der beiden folgenden Aufgaben aus Sicht des Anwendungsentwicklers:

- Formalisierung des Optimierungsproblems. Diese Aufgabe erfordert die detaillierte Kenntnis von Fachkonzepten, Benutzermodellen etc. und gehört daher zur Entwicklung der Anwendungslösung.
- Entwicklung und Bereitstellung von Optimierungsverfahren. Diese Aufgabe wird durch das Entwicklungssystem ConSolve erledigt.

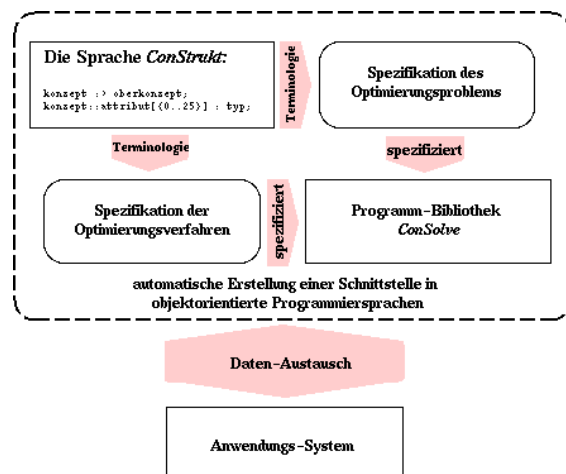
Bindeglied dieser beiden Aufgabenstellungen ist die deklarative Formalisierung des Optimierungsproblems in der Beschreibungssprache ConStrukt.

Kombinatorische Optimierungsprobleme sind im Allgemeinen nicht effizient lösbar. Es gibt eine Reihe von Suchverfahren, die alle ihre Vor- und Nachteile haben. Das ConSolve-System stellt momentan eine Reihe sog. constraint-basierter Verfahren zur Verfügung, die ggf. in lokale Suchalgorithmen eingebettet werden. Aus diesen wählen Anwendungsentwickler ein geeignetes Verfahren aus.

Das Anwendungssystem verwendet dann zur Lösung

der formalisierten Planungsaufgabe die Laufzeitbibliothek von ConSolve. Zur Kommunikation zwischen Anwendungssystem und Laufzeitbibliothek wird aus der Problemformulierung eine Schnittstelle in einer höheren Programmiersprache automatisch generiert. Momentan wird dabei C/C++ unterstützt, die native Unterstützung der Programmiersprache Java ist in Planung.

Die folgende Abbildung beschreibt das Vorgehensmodell bei der Verwendung von ConSolve.



Dieses Vorgehen bietet auch bei der Pflege und Erweiterung der mittels ConSolve entwickelten Systeme folgende Vorteile:

- Die kompakte Problemspezifikation erleichtert die Anpassung an veränderte Gegebenheiten durch Einfügung oder Streichung von Randbedingungen oder veränderten Bewertungen.
- ConSolve wird fortlaufend um neue Verfahren erweitert. Diese lassen sich aufgrund der Deklarativität leicht durch bereits bestehende Formalisierungen von Optimierungsproblemen nutzen.

Beschreibungssprache ConStrukt

Das ConSolve[®]-System löst kombinatorische Optimierungsprobleme. Dazu werden die Problemstellungen in der speziellen Beschreibungssprache **ConStrukt** formuliert und dann dem ConSolve-Laufzeitsystem übergeben.

Die Sprache ConStrukt wurde entworfen, um logische und kombinatorische Probleme möglichst einfach formulieren zu können ("domänenspezifische Sprache"). Auf diese Weise kann die Formulierung sehr nahe an der Problemstellung gehalten werden, sie ist übersichtlich und die Arbeit daran wenig fehlerträchtig. Darüber hinaus ist es möglich, relativ schnell zu einer Formulierung für ein Problem zu gelangen.

In der Sprache ConStrukt werden keine Lösungswege beschrieben, sondern lediglich Problemstellungen formuliert. Die Auswahl der zur Problemlösung verwendeten Algorithmen geschieht durch das ConSolve-Laufzeitsystem. Somit handelt es sich zugleich um eine "deklarative Sprache".

Anhand des N-Damen-Problems soll die Sprache ConStrukt beispielhaft eingeführt werden: Positioniere n Damen eines Schachspieles so auf einem Spielbrett der Größe $n \times n$, dass sich die Damen nicht gegenseitig schlagen können.

```
01 // Das N-Damen-Problem in ConStrukt
02 export Damebrett, Groesse > c.cpp;
03
04 // Erlaubte Werte für Parameter
05 Groesse : {4..100};
06
07 // Verwendete Brettgroesse
08 Groesse := 8;
09
10 concept Damebrett
11 : [_[{1..Groesse:Z}] : {1..Groesse :Z}];
12 begin
13   entail
14     forall spalte:{1..Groesse :Z}
15       ensure alldiff this[spalte];
16   entail
17     forall spalte:{1..Groesse :Z}
18       ensure alldiff this[spalte] + (spalte-1);
19   entail
20     forall spalte:{1..Groesse :Z}
21       ensure alldiff this[spalte] - (spalte-1);
22 end Damebrett;
```

Über "export... > c.cpp" werden die Konzepte "Damebrett" und "Groesse", die im weiteren Verlauf des Beispiels eingeführt werden, für den Zugriff aus einer C/C++-Anwendung heraus freigegeben. Die Lösungen können über einen sogenannten Iterator aus dem Anwendungssystem heraus aufgezählt werden. In unserem Beispiel kann zusätzlich die Größe des Spielbrettes zur Laufzeit festgelegt werden.

Das Spielbrett wird im Konzept "Damebrett" in Zeile 11 als eindimensionales Feld von Spalten definiert. Jede Variable definiert die Zeilenposition einer Dame in der entsprechenden Spalte.

Die eigentliche Problemstellung wird im Konzept "Damebrett" in den Zeilen 14 bis 22 definiert. Um das N-Dame-Problem zu erfüllen darf sich

1. in jeder Spalte nicht mehr als eine Dame befinden,
2. in jeder Zeile nicht mehr als eine Dame befinden und
3. ausgehend von jeder Dame keine weitere Dame auf den Diagonalen befinden.

Die erste Bedingung ist bereits durch die Modellierung des Problems im Konzept "Brett" erfüllt. Die zweite Bedingung wird durch die Zeilen 17 bis 19 sichergestellt. Die dritte Bedingung wird durch die Zeilen 20 bis 22 festgelegt.

Das N-Damen-Problem ist nun mit der obenstehenden Problemformulierung in der Sprache ConStrukt skalierbar beschrieben und kann in eine Anwendung integriert werden.

Integration

Das ConSolve[®]-Laufzeitsystem stellen wir Ihnen als dynamische Bibliothek zur Verfügung, die Sie in jede Programmierumgebung mit C- oder C++-Schnittstelle einbinden können.

Maximalen Komfort und hohe Produktivität bietet unsere elegante Programmierschnittstelle:

Hierzu stellen wir ConStrukt-Konzepte für den C++-Zugriff zur Verfügung, indem über Codegenerierung aus der Problembeschreibung C++-Klassen und Methoden erzeugt werden. Die notwendigen Aufrufe der Laufzeitbibliothek erfolgen im generierten Code und bleiben für den Anwendungsentwickler verborgen.

Neben den Standardfunktionen der Programmierschnittstelle wie z.B. das Aufzählen von Lösungen über Iteratoren oder die Auswahl eines Suchverfahrens, stellt die Programmierschnittstelle eine Reihe von erweiterten Funktionen zur Verfügung:

Während der Suche nach einer Lösung kann es erforderlich sein, das Anwendungssystem über den Zustand des Suchvorganges laufend zu informieren und ihm gleichzeitig einen Eingriff in den Suchvorgang zu ermöglichen. Hierzu werden Beobachter-Klassen - benannt nach dem verwendeten Entwurfsmuster - zur Verfügung gestellt, die den Informationsfluss zwischen Laufzeitsystem und Anwendung gewährleisten. Beispielsweise kann ein Beobachter für eine Fortschrittsanzeige oder für den vorzeitigen Abbruch eines Suchvorganges eingesetzt werden.

Häufig ist es erforderlich, eine in ConStrukt formulierte Problembeschreibung während der Laufzeit einer Anwendung zu modifizieren. Hierzu können Sie die Programmierschnittstelle nutzen, indem das Anwendungssystem zusätzliche Bedingungen einer Problembeschreibung hinzufügt.

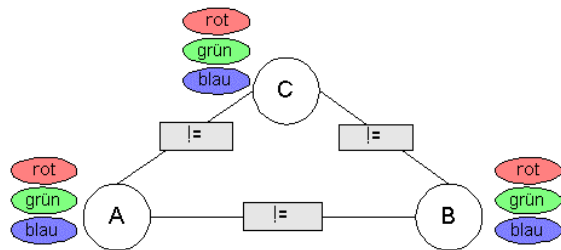
Sofern eine Problembeschreibung Bedingungen unterschiedlicher Priorität enthält, die gemeinsam nicht alle erfüllbar sind, ist es wünschenswert, die Qualität der berechneten Lösungen darstellen zu können. Über die Programmierschnittstelle können Sie die durch eine berechnete Lösung verletzten Bedingungen abfragen, beispielsweise um die Lösungsqualität dem Benutzer einer Anwendung anzuzeigen.

Exkurs: Constraint-Techniken

In ConSolve® werden Optimierungsprobleme als Auswahlprobleme spezifiziert. Eine Lösung des Problems muss also für jede Variable aus ihren alternativen Belegungen genau eine zulässige Belegung auswählen. Hierbei dienen Bedingungen (Constraints) dazu, die Auswahl einzuschränken.

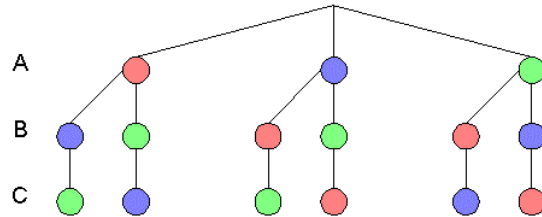
Die Zielfunktion bewertet die Qualität einer Variablenbelegung und identifiziert dadurch Lösungen des Optimierungsproblems. Bei ConSolve ergibt sich die Zielfunktion aus der Bewertung der definierten Bedingungen. Harte Bedingungen sind von einer Lösung des Optimierungsproblems unbedingt zu erfüllen. Gewichtete und unscharfe Bedingungen müssen von einer möglichen Lösung nicht zwangsläufig erfüllt werden, gehen jedoch in die Zielfunktion ein.

Zur Lösung eines Problems durchsucht ConSolve den Lösungsraum (Suchraum), indem Variablenbelegungen aufgezählt werden. Hierbei wird ein sogenannter **Suchbaum** aufgebaut. Während der Suche werden die Bedingungen ausgewertet (Propagierung), wodurch Variablenbelegungen, die keine Lösungen darstellen, frühzeitig erkannt und aus dem Suchbaum entfernt werden. Erst die Kombination dieser beiden Verfahren führt bei größeren Optimierungsproblemen zu einem akzeptablen Zeitverhalten. Die Suche nach einer Lösung soll anhand eines Beispiels erläutert werden



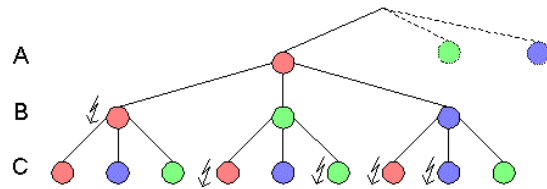
Die obige Abbildung zeigt die Variablen A, B und C, die jeweils mit einer Ungleich-Bedingung miteinander ver-

bunden sind. Jede der Variablen besitzt den Wertebereich "rot", "grün" und "blau". Bei der Suche werden Bedingungen ausgewertet, wodurch unzulässige Variablenbelegungen sofort erkannt werden. Hieraus ergibt sich folgendes Suchbaum, der sechs Lösungen enthält:



Beispielsweise wird nach dem Belegen der Variable A mit dem Wert "rot" gleich erkannt, dass die Variable B nicht ebenfalls mit "rot" belegt werden kann (Ungleich-Bedingung zwischen A und B), wodurch der entsprechende Zweig aus dem Suchbaum gestrichen werden kann.

Wenn wir alternativ eine vollständige Baumsuche ohne Propagierung der Bedingungen durchführen, ergibt sich ein viel größerer Suchbaum aus 27 Variablenbelegungen. Die folgende Abbildung zeigt einen Ausschnitt dieses Suchbaums, bei dem ein Zweig (Variable A wird mit "rot" belegt) vollständig durchlaufen wird.



Hier ist zu erkennen, dass innerhalb des Zweiges sieben unzulässige Lösungen aufgezählt werden, die jeweils zu einem Rücksprung im Baum (Backtracking) führen.

Produktmerkmale

Entwicklungssystem

- Die deklarative, domänenspezifische Sprache ConStrukt dient zur Modellierung der Problembeschreibung.
- Die Codegenerierung von problemspezifischen Schnittstellen zum Laufzeitsystem ermöglicht einfache und schnelle Integration in Anwendungen.
- Eine spezielle Shell ermöglicht interaktives Arbeiten ohne Anwendungsprogramm.
- Ein Werkzeug zur Analyse von Suchläufen ist vorhanden.

Laufzeitumgebung

- Das effiziente Laufzeitsystem zur Lösungssuche implementiert verschiedene Constraint-basierte Techniken.
- Zwischenstände der Lösungssuche können abgerufen werden.
- Änderungen an der Problemstellung zur Laufzeit werden unterstützt.
- Die Analyse von Zwischenergebnissen oder extern erstellten Lösungsvorschlägen ist möglich.
- Während der Lösung ist ein Abbruch jederzeit mit einem Zwischenergebnis möglich.

Constraint-Techniken

- Symbolische Wertebereiche und Mengen von Zahlen (ganze Zahlen, Fließkommazahlen) können verarbeitet werden.
- Harte, gewichtete und unscharfe Constraints (Fuzzy-Constraints) werden unterstützt.
- Verschiedene vordefinierte globale Constraints sind verfügbar.
- Die folgenden Suchalgorithmen sind verfügbar:
 - § Branch&Bound Baumsuche mit oder ohne Backmarking.
 - § Programmierbare lokale Suche.
- Die folgenden Formen der Constraint-Propagierung werden unterstützt:
 - § Extended forward checking.
 - § Inverse arc consistency (Fixpunkt der Propagierung).
 - §

Systemvoraussetzungen

- Das ConSolve[®]-Entwicklungssystem und die Laufzeitumgebung unterstützen folgende Betriebssysteme:
Microsoft Windows 9x , ME, NT, 2000 oder XP;
Linux und UNIX (Apple Macintosh in Vorbereitung).
- Das ConSolve[®]-Laufzeitsystem lässt sich in jede Programmierumgebung mit C- oder C++-Schnittstelle einbinden.



<http://www.consolve.de>

SIEDA GmbH
Richard-Wagner-Straße 91,
67655 Kaiserslautern
Deutschland

Telefon: +49 631 363015 0
Fax :+49 631 383015 33
E-Mail: info@sieda.com