Modeling Languages and Global Optimization

Languages and Conceptual Design Features

Hermann Schichl

Universität Wien

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Contents

- Modeling
- History of Modeling Languages
- Modeling Languages
  - Features
  - Advantages
- Global Optimization / Verified Computing
- Gl.Opt. and Modeling Languages
- Important Classes of Problems
- Modeling Languages in the Future

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

The COCONUT project

- European Union research and development project
- Partners from six European universities:
  Nantes, Lausanne, Vienna
  Louvain-la-Neuve, Coimbra, Darmstadt
  and an industrial partner:
  ILOG
- Aimed at the integration of the existing approaches to continuous global optimization and constraint satisfaction
- December 2000 – November 2003
- Uses AMPL as modeling language, also a GAMS interface
- http://www.mat.univie.ac.at/coconut

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Modeling

- From the Latin word "**modellus**", typical human way of coping with the reality
- Mechanical models
  - Astronomy
  - Architecture
- Mathematical models used since Hellenic Age
  - Geometry (~600 BC (Thales) — ~300 BC (Euclid))
  - Circumference / Diameter of Earth (~250 BC (Erathostenes))
  - Arithmetic (~240 AD (Diophantus) — ~800 AD (Al-Khwarizmi))
  - Astronomy (~150 AD (Ptolemy))
  - Physics
  - Economics
- Visual models
  - Anatomy

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Modeling (ctd.)

- A model is a **simplified version** of something that is real.
- Represents a **real-world object** by other, simpler objects.
- **Structures** the knowledge about the real world.
- **Reduction** to those phenomena and aspects which are considered important.
- Usefulness is restricted to its **scope of application**.
- Serve many **different needs** ≈ the **modeling goal**
  - explain phenomena, make predictions,
  - control environment, decision making,
  - maintenance, normative components,
  - communication

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

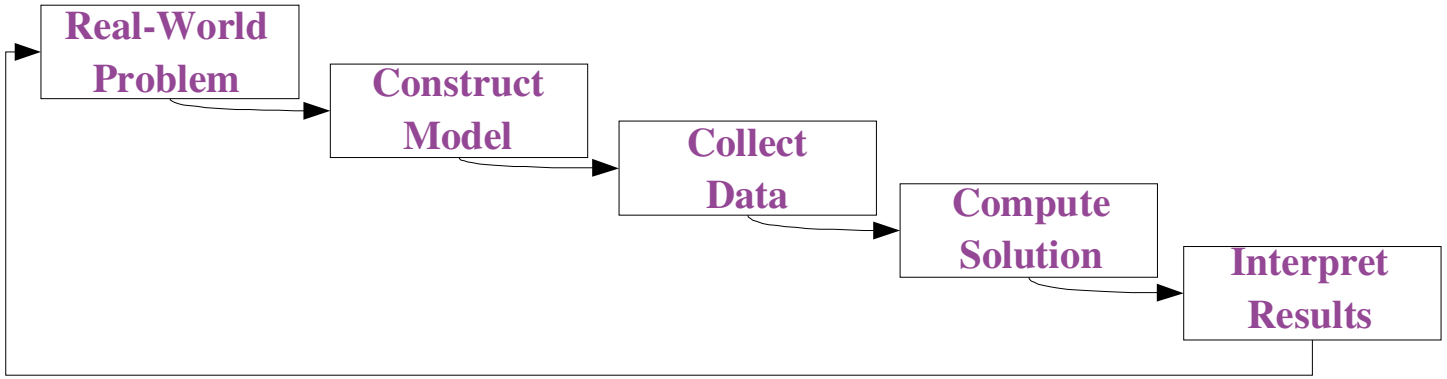
Mathematical Modeling

- Represents a **real-world problem** by mathematical objects in a formalized mathematical language.
- Can be analyzed by **mathematical theory** and **algorithms.**
- Represents the knowledge about the real-world by **concepts**, **variables**, **relations**, **data**.
- Makes models accessible to **computers**.
  - First computers were human (usually underpaid women).
  - Since ENIAC (1945) larger and more complicated mathematical models have become accessible.
- Models became **attractive** to **military** and **industry**.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Modeling – Scheme

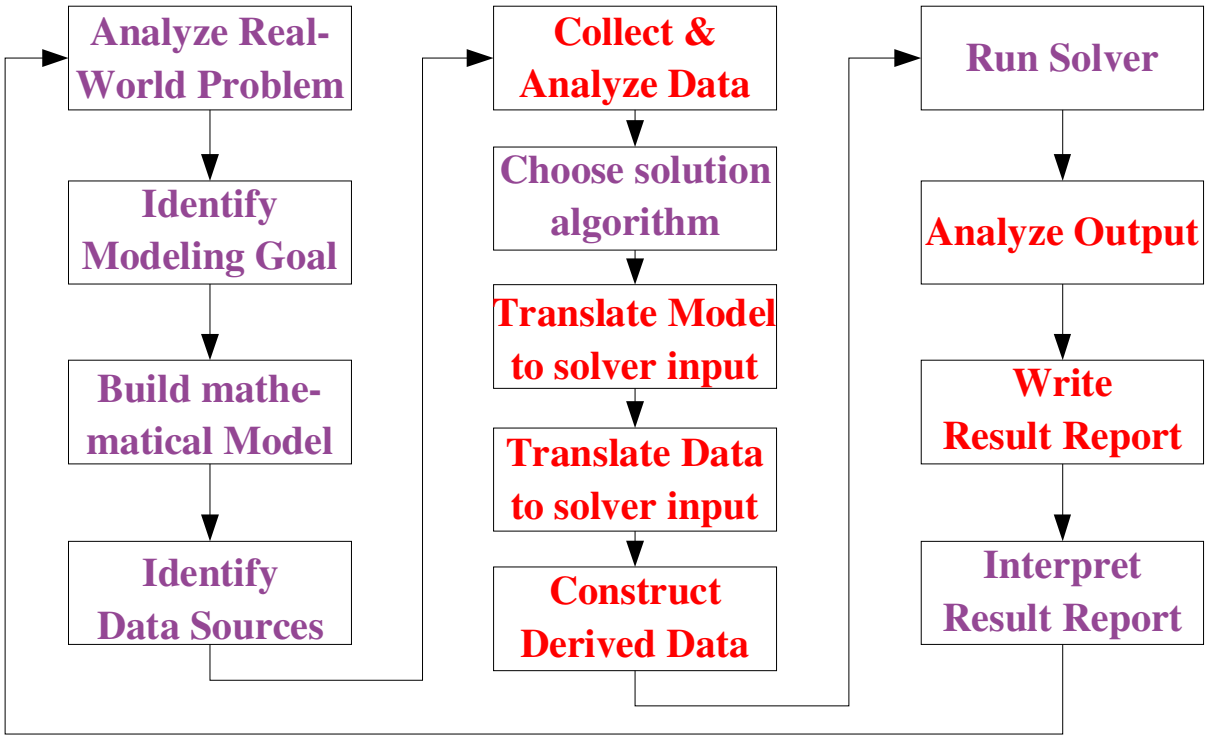


- Modeling is done in "cycles".
- Some stages need the help of the end-user.
- For complex models, a computer is needed in the Compute Solution step.
- The Collect Data step is nowadays computer based, as well.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Modeling – Detailed Scheme



IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

History of Modeling Languages

- Mathematical Modeling is used in philosophy and scientific communities since antiquity.
- Useful **algorithms** (e.g. Runge-Kutta, FFT) made model analysis "tractable".
- The development of **computers** made bigger models accessible.
- Problems were **input** as **programs**.
- Data was read in from files (which were hand-made)
- Derived information (e.g. derivatives) were **programmed**.
- This had many disadvantages: **error-prone**, **unstructured**, **not maintainable**, **no re-usable**, **not flexible**, **badly scaling**

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

History of Modeling Languages (ctd.)

- In 1947 George Dantzig developed the simplex algorithm for solving **linear programming** problems for the US air force.
- LP was used by **non-scientific** users to optimize various tasks.
- **Modeling** sometimes was more **expensive** than the profit.
- For LP mainly **matrices** were needed.
- The **MPS** input format was invented (for MPSX by IBM).
- For translating data to MPS or another input format, computers were used — **matrix generators**. (PDS, MAGEN, OMNI)
- Then **MGG** (matrix generator generator) was developed.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

History of Modeling Languages (ctd.)

- Even then the **Modeling process** was still **expensive**.
- Modeling was difficult and needed expert knowledge, modelers came from **very different areas**, and had very **different background knowledge**.
- Many self-devised modeling systems were (and still are) written.
- **Optimization** was most interesting for the industry.
- Late 1970s (papers in 1980) **GAMS** was the first algebraic modeling language.
- Late 1980s **SIF** as input language to Lancelot (paper 1990),
- Around 1985 **AMPL** was developed (book 1992),
- **Xpress-MP** — Dash Optimization, 1987

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

History of Modeling Languages (ctd.)

- Around the 1990s the number of modeling systems increased.
- LINGO — LINDO Systems; Chicago; 1988
- MPL — Maximal Software; 1989
- AIMMS — Paragon; 1993
- ECLiPSe — IC-Parc; London; 1995
- LPL — Hürlimann; Zürich; 1995
- NOP, NOP-2 — Neumaier, Dallwig, Schichl; Vienna; 1997-1999
- Numerica — Van Hentenryck, ILOG; Paris; 1997
- OPL — ILOG; Paris; 1998
- MINOPT — Schweiger, Floudas; Princeton; 1999
- Mosel — Dash Optimization; 2002

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Modeling Languages – why?

- Make modeling **convenient** and **reduce** the probability for **errors**,
- **Reduce** modeling **costs**,
- Increase the **flexibility** of the models,
- **Independence** from **solvers**,
  - Automatically translate models to internal solver formats.
  - Different solvers can be tried on hard problems
- **Independence** from **data sources**.
  - Automatically retrieve data and convert them to solver formats.
- Automatize the error-prone generation of **derived information** like derivatives
  - Automatic differentiation

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Algebraic Modeling Languages

- **Declarative Languages** — neither imperative, functional nor logic programming
- Models are expressed in an **index-based** formulation.
- Typical: **Sets**, **Indices**, **Parameters**, **Variables**
- Conceptual similar entities are grouped in **Sets**.
- These entities are later referenced by **indices** to the elements of these sets.
- Groups of entities (variables, constraints,...) can be represented in a compact by one **algebraic expression**.

$\sum_{i \in X} x_i$ is represented in AMPL by the expression

```
Sum {i in X} x[i];
```

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Algebraic Modeling Languages (ctd.)

- The algebraic modeling language is responsible for creating a **problem instance** to work on (set-indexing ability).
- Capability of dealing with **non-linear** models.
- The **notation** is close to the **mathematical** formulation.
- Problem formulation can be **data-independent**.
- Model is **scalable**, so switch from **toy-problem** to **full problem** can be done by a simple change of the parameters.
- Most **statements** are **declarative**, very **few procedural** statements, except for **conditionals and loops**.
- Many provide a **database interface** for data retrieval.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Algebraic Modeling Languages (ctd.)

mathematical model

$$\begin{aligned} &\min\ x^T Q\, x + c^T x \\ &s.t.\quad Ax \le b \\ &\qquad \| x \|_2 \ge 1 \\ &\qquad x \in [\, l\,, u\,] \end{aligned}$$

"flat" model

$$\begin{aligned} &\min \sum_{i=1}^{N} \sum_{j=1}^{N} Q_{ij}\, x_i\, x_j + \sum_{k=1}^{N} c_k\, x_k \\ &s.t.\quad \sum_{j=1}^{N} A_{ij}\, x_j \le b_i \qquad \forall\, i = 1,\ldots, M \\ &\qquad \sum_{j=1}^{N} x_j^2 \ge 1 \\ &\qquad x_j \in [\, l_j\,, u_j\,] \qquad \forall\, j = 1,\ldots, N \end{aligned}$$

AMPL model

```
### PARAMETERS ###
param N>0 integer;
param M>0 integer;
param c {1..N};
param b {1..M};
param Q {1..N,1..N};
param A {1..M,1..N};
param l {1..N};
param u {1..N};
### VARIABLES ###
var x {1..N};
### OBJECTIVE ###
minimize goal_function:
  sum{i in 1..N}(sum{j in 1..N} Q[i,j]*x[i]*x[j])+
     sum{j in 1..N} c[j]*x[j];
### CONSTRAINTS ###
subject to linear_constraints {i in 1..M}:
   sum {j in 1..N} A[i,j]*x[j] <=b[i];
box_constraints {j in 1..N}: l[j]<=x[j]<=u[j];
con_add:   sum { j in 1..N} x[j]^2 >= 1;
#####################################
data HQP2.2.dat;
solve; display X;
```

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Non-algebraic modeling languages

- **Object oriented modeling languages**
  - structured modeling (primitive entity, compound entity, attributes, tests, functions)
  - gPROMS, ASCEND for chemical engineering
  - EXTEND for manufacturing processes
- **Languages for constraint logic programming**
  - ECLiPSe
  - provide a model description with very specialized constraints (all_diff, count, ...)
  - hints on the solution process, preferred branching and backtracking points can be specified

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Non-algebraic modeling languages (ctd.)

```
% ECLiPSe SAMPLE CODE
% AUTHOR:       Joachim Schimpf, IC-Parc
% The famous N-queens problem
:- set_flag(gc_interval, 100000000).
:- lib(lists).
:- lib(fd).
:- lib(fd_search).
%--------------------
% The model
%--------------------
queens(N, Board) :-
 length(Board, N),
  Board :: 1..N,
   ( fromto(Board,[Q1|Cols],Cols,[]) do
    ( foreach(Q2,Cols),param(Q1),
              count(Dist,1,_) do
      noattack(Q1,Q2,Dist)
    )
   ).
noattack(Q1,Q2,Dist) :-
        Q2 #\= Q1,
        Q2 - Q1 #\= Dist,
        Q1 - Q2 #\= Dist.
```

```
%-----------------------
% The search strategies
%-----------------------
labeling(a, AllVars) :-
 ( foreach(Var, AllVars) do
    count_backtracks,
    indomain(Var)      % select value
 ).
labeling(b, AllVars) :- % also for c,d,e
 .... % stuff deleted here ! %
first_queens(Strategy, N) :- % Find one solution
  statistics(times, [T0|_]),
  queens(N, Board),
  statistics(times, [T1|_]),
  D1 is T1-T0,
  printf("Setup for %d queens done")
  init_backtracks,
  labeling(Strategy, Board),
  get_backtracks(B),
  statistics(times, [T2|_]),
  D2 is T2-T1,
  printf("Found first solution for %d queens in %w\
          s with %d backtracks%n",[N,D2,B]).
% further stuff deleted down there...
```

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Other Modeling Tools

- **Integrated Modeling Environments**
  - Include **visualization techniques**.
  - Graphical User Interface (**GUI**) for model management and result presentation
  - OPLStudio, AIMMS
- **Model Analysis Tools**
  - **analyze mathematical problems** wrt.
    constraint effectiveness checking, objective effect analysis, constraint grouping/sorting, shape analysis,...
  - ANALYZE (linear), MPROBE (non-linear)
- **Spreadsheet modeling**
  - FRONTLINE Systems provide optimization on EXCEL **spreadsheets**.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Advantages of Modeling Languages

- Scalability of the model
- Clear cut between modeling and numerics
- Clear cut between model structure and data
- Independence of solvers and solution algorithms
- Derived information can be generated automatically
  - notably automatic differentiation tools are widely used to provide derivatives and sparse Hessians
- The error probability in the translation of model and data to the solver's internal representation is almost zero.
- Convenience for the user.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Important Features of Modeling Languages

- Close to mathematical Notation
- Little overhead in the formulation
- Convenience for the modeler
  - Names, arbitrary index sets,...
- Flexibility in the type of models
- Simple interface for solver connection
  - Modeling language to solver (model, data, derived information)
  - Solver to modeling language (progress report, solution, status info.)
- Simple and powerful data handling
  - Additional files
  - Database connectivity
- Automatic differentiation

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Global Optimization/ Validated Computing

- Various levels of model rigor can be distinguished

  Validation ≤ Verification/Falsification ≤ Mathematical Proof

- **Rigorous Computing** tries to provide computer assisted mathematical proofs.
- **Deterministic global optimization** tries to solve

$$\begin{aligned} &\min f(x) \\ &s.t. \quad F(x) \in \boldsymbol{F} \\ &\qquad x \in \boldsymbol{X} \end{aligned}$$

  in a rigorous way.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Global Optimization (ctd.)

- The variables typically fall into one of the following categories:
  - **Continuous variables** (real or complex valued), usually **bounds** are provided for them
  - **Integer variables** (also real or complex) usually **bounds** are specified in addition
  - **Binary variables** (either ***0*** or ***1***)
  - **Discrete variables** (can take a finite number of distinct isolated values) usually these values can have arbitrary symbols
  - **Semi-continuous variables** (***0*** or **at least some threshold**)
  - **Partial integer variables** (either ***0***,...,N or continuous **bound constrained in** $[N, M]$)

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Global Optimization (ctd.)

- The constraints $F(x)$ also come in different flavors
  - **Continuous constraints** (linear, convex, non-convex)
  - **Discrete constraints**
  - **Logical constraints**
    involving binary variables and logical operators
  - **Global constraints**
    constraints involving a lot of variables (e.g. all_different, cardinality)
  - **Special ordered sets**
    SOS-1: At most one of a set of variables is non-zero,
    all the others are zero
    SOS-2: At most two 'adjacent' variables are non-zero,
    all the others are zero.
  - **Exclusion Regions**
    $x_L \notin X_L$

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Global Optimization (ctd.)

- The data sometimes needs to be **known exactly**. Roundoff in the translation from modeling system to solver can destroy important properties or change the solution set, e.g.

  $0.4\,x + 0.6\,y \geq 1$
  $x\,, y \in [\,0\,, 1\,]$

  becomes infeasible in IEEE floating point arithmetic after rounding the data.
- Sometimes the data is **uncertain**.
- Some constraints may be less relevant than others, or they can be slightly violated in some circumstances (**soft constraints**).
- **Structural information** about the problem should not be lost in the translation from modeling system to solver, and should be specifiable in the first place.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Typical Problems

- Most modeling languages have originally been designed for non-linear **local optimization**.
- They pass **presolve**d problems to the solver. Most of the problem's mathematical structure is lost (except for sparsity patterns).
- The data and the coefficients are passed as **floating-point numbers** (this involves roundoff and makes mathematical rigor impossible).
- Uncertain data cannot be specified.
- There is only restricted support for soft/hard constraints.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Typical Problems (ctd.)

- There is **no** inherent **support** for **interval arithmetic**, so important derived information (such as implied bounds) cannot be generated automatically (except in Numerica).
- Usually there is no support for **matrix operations**.
  $A^T D A x = b$ usually can be used in a model, but with difficulty, destroying the matrix structure.
  It is close to impossible to specify the constraint $det(A) \geq c$.
- Finally, many difficult global optimization problems can only be solved by using different but mathematically **equivalent formulations** of the same model.
  No modeling language (I know of) can be used to specify that.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Other classes of Optimization Problems

- Multi-level Optimization
- Multi-objective Optimization
- Optimization with many objective functions
- Safety Programming
- Resource Optimization
- Stochastic Optimization
- Optimization involving differential equations
- ... and (hierarchical) combinations of all above mentioned classes

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Constraint Satisfaction Problems

- **No objective function**, only sets of constraints
- **Various goals** are **possible**
  - find one solution (e.g. map-coloring problem)
  - find all solutions (in case of isolated solution points)
  - find outer and inner coverings of the solution set to provide a visualization
  - find N sufficiently distinct solutions
  - prove that the constraints are inconsistent
- Perform all the above **approximately or with full rigor**.
- It should be possible to formulate these goals in the modeling language.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Multi-Objective Optimization

- **More than one objective** is present
- Search for **Pareto-optimal** solutions
  - find one
  - find all
  - cover the extremal set
- Multiple objectives should be allowed in a modeling language.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Optimization with multiple objective functions

- **More than one objective function** is specified.
- **For every objective** function the optimization problem is **solved**, the results are recorded independently
- This should not be handled by simply passing each problem instance independent of the other to the solver.
- Support for **warm starts** and **parallel execution**.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Multi-Level Optimization

- **Hierarchically structured** optimization problems
- The simplest ones are **minimax problems**
- A two stage problem:
  The inner (stage 1) problem consists of g, G, y and has **free variables** x which appear in stage 2 together with f, F.

$$
\begin{aligned}
&\min f(x) \\
&s.t.\quad F(x) \in \boldsymbol{F} \\
&\qquad \max g(x, y) \le c \\
&\qquad s.t.\quad G(x, y) \in \boldsymbol{G} \\
&\qquad\qquad y \in \boldsymbol{Y} \\
&\qquad x \in \boldsymbol{X}
\end{aligned}
$$

- Recursive problem formulations would be useful (i.e. a model with parameters should be usable as a function)

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Safety Programming

- Verification of **safety constraints for uncertain designs**
- This is e.g. needed for mechanical engineering problems: Show that (or find a counterexample)

$$C(x,u) \le 0$$
$$\forall\, x,u : F(x,u) = 0$$
$$\|x - x_0\| \le \Delta x$$

- A typical problem of this type is e.g. FEM structural analysis, where $F(x,u)=0$ takes the special form

$$A(x)u = b$$

and the x are bound constrained.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Resource Optimization

- Optimization problems from resource optimization tend to be very large and very structured.
- All constraints can be generated from building blocks
- All data can be retrieved from databases
- There is a hierarchical structure of problems of increasing complexity
- Constraints for different resources are only similar.

$$\boldsymbol{B}(z_1,\dots,z_q)=\sum_{i\in I}\boldsymbol{b}_i(s_i,z_1,\dots,z_q)$$

$$p\in\mathbb{R}(\mathbb{Z}),\quad s_i\in\mathbb{R}(\mathbb{Z})\ \ \forall i\in I$$

$$z_j\in\mathbb{R}(\mathbb{Z})\ \ \forall j=1,\dots,q$$

$$y_n\in\{0,1\}\ \ \forall n\in E$$

$$s_i=f_i(p)\ \ \forall i\in I_0\subset I$$

$$s_i=S_i(p,z_1,\dots,z_q)\ \ \forall i\in I_p$$

$$s_i\le 0\ \ \forall i\in I_{\text{in}}\subset I\setminus(I_0\cup I_p)$$

$$s_i\ge -M_i\,y_{n_i}\ \ \forall i\in I_{\text{in}}\cap I_d$$

$$s_i\ge 0\ \ \forall i\in I_{\text{out}}\subset I\setminus(I_0\cup I_p)$$

$$s_i\le M_i\,y_{n_i}\ \ \forall i\in I_{\text{out}}\cap I_d$$

$$g_j(p,s_{j_1},\dots,s_{j_n})\le 0\ \ \forall j\in J_u,j_1,\dots j_n\in I$$

$$h_k(p,s_{k_1},\dots,s_{k_m})=0\ \ \forall k\in J_g,k_1,\dots k_m\in I$$

$$G_j(z_1,\dots,z_q)\le 0\ \ \forall j\in J_u^z$$

$$H_k(z_1,\dots,z_q)=0\ \ \forall k\in J_g^z$$

$$\sum_{i=1}^{m_r}y_{n_i}\le e_r\ \ \forall r\in R$$

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Optimization involving differential equations

- In **design problems** (e.g. aircraft design) often constraints are defined using the solution of some **ordinary** or **partial differential equation**.
- Sometimes **differential algebraic systems** and **integral equations** have to be used in the modeling process (e.g. parameter optimization for ovens used in silicon wafer production)
- In very few modeling systems (e.g. PCOMP, MINOPT) exists the possibility to specify differential equations.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Stochastic Optimization

- Many **real-world problems** (e.g. portfolio optimization) depend on **stochastic processes** (e.g. the stock market)
- In addition to the before-mentioned variable classes **random variables** enter the mathematical problem formulation.
- Stochastic variables can be **continuous or discrete**.
- Usually **distribution functions** and/or **covariance matrices** are available.
- Often stochastic processes are included in the model. This leads to multistage problems.
- **New** classes of **constraints**, involving **expectation values**, **variances**, **covariances**, or **probabilities** appear.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Stochastic Optimization (ctd.)

- A typical stochastic optimization problem has the form

$$\begin{aligned} &\min f\left(x^{(k)}, \xi^{(k)}\right) \\ &s.t. \quad F^{(k)}\left(\xi^{(k)}, x^{(<k)}, \xi^{(<k)}\right) \in \boldsymbol{F} \\ &\qquad x^{(k)} \in \left[x_0^{(k)}\right] \\ &\qquad \xi^{(k)} \sim g^{(k)}(b, x) \end{aligned}$$

  where the k-th stage variables $x^{(k)}$ and $\xi^{(k)}$ depend on the lower stages in this multi-stage problem. The constraint $F^{(k)}$ usually involves expectations or probabilities.

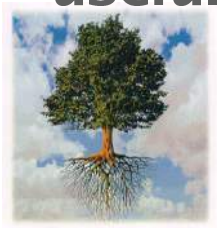
- Today's algorithms use **scenarios** to "solve" this problem, they are, however, not part of the model.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Beyond Stochastic Optimization

- Stochastic optimization problems are still not the most general ones.
- The modeler wants to record his knowledge about the real-world in the modeling language to construct the **model**.
- Typical knowledge is e.g.:
  - the opinion of expert(s)
  - within certain bounds
  - the mean value (and variance, higher moments)
  - a distribution function
  - a set of discrete values (with or without probabilities)
  - an exact value, an approximation, an estimate, a guess
- Then all objects must be set in relation, and these relations can be hard or soft or stochastic,...
- It should be possible to build all mathematically equivalent useful formulations.



IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Data Handling

- In addition to the declarative part, the **model instance** must be specified by adding data.
- Knowledge about the data should be recordable
  - Source of the data and how much it suits the model
  - Accuracy of the data
  - Is the data up-to-date?
  - Validity of the data
- Retrieval and structuring the data should be convenient
  - Database (ODBC) interface
  - Data/Model separation should be possible

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Beyond Stochastic Optimization (ctd.)

- In addition to the model instances, sometimes solution hints must be used to guide the solvers.
- This is neither part of the model nor part of the model instance, and includes e.g.
  - search and solution strategies
  - scenarios for stochastic problems
  - starting points for local optimization
  - bisection and backtracking hints
- Different solvers will need different hints.
- Construct a solver's **view of the model instance**.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Visions for the Future

- Matrix support
- Allow several equivalent formulations at the same time
- GUI front-ends
- Object-oriented
- Recursive
- Building blocks
- Possibility to specify all knowledge

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

GUI front-ends and back-ends

- Like in the Mathematica 4 GUI front-end provide an even **more mathematical notation**.
- Provide **model structuring**
  - coarse grain / fine grain view
- Provide **syntax-based editing** capabilities with on line-help
- Visualization of the connection between model-parts
- Cross reference: variables – constraints
- **Progress** visualisation
- **Result** visualisation
- Is already done in some systems (e.g. AIMMS, Mosel, OPLstudio, MPL)

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Object Oriented Modeling

- Object oriented programming is an extension (or the consequence) of imperative programming
- Modeling needs the consequence of declarative programming — **derived models**.
  - E.g. a screw is represented by a set of variables and constraints.
  - Steel is a material, also specified by variables and constraints.
  - Wood is another material.
  - Now it should be straight-forward to derive steel screws and wooden screws by inheritance
- Mimics the mathematical way of defining new objects.
  - E.g. a group is a monoid where inverses of all elements exist.
  - Associativity and unit element are “inherited” from the monoid.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Recursive Modeling

- **Multistage problems** often arise naturally.
- E.g. first we want to prove that a designed robot is free of structural deficiencies.
- We build a model for that (a safety problem) and solve it.
- For the next design we want the cheapest robot which is free of structural deficiencies.
- This is a two-stage optimization problem (a design problem) where the first stage contains the safety problem which has been designed already. However, now the design parameters are no longer fixed but free variables).
- **Models should be reusable** as "functions" **in bigger models.**

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Building Blocks

- Many huge optimization problems are built from a **big number** of (almost) **identical blocks**
  - chemical engineering
  - resource optimization
  - manufacturing processes
  - optimal scheduling
- Supporting building blocks in modeling languages and enhancing it by adding GUI support will **decrease the number of modeling errors** for big problems and greatly reduces the modeling costs.
  (e.g. ProComm BoFit)

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Specifying all knowledge

- It should be possible for the modeler to **record all** his **knowledge** about model and real-world problem in the language.
- Knowledge about objects
- Knowledge about data
- Knowledge about solvers and solution strategies
- The solver need not take all information into account if it cannot make use of certain information.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Open Model Exchange Format/ Model Databases

- An open format for communicating models, storing them in databases and exchanging them between modeling languages/systems would be very useful.
  - In the COCONUT project we had to write >10 converters between various modeling languages for testing the same set of benchmark problems on a number of different solvers.
- Could be XML like and would provide a convenient Web-interface for model presentation at the same time,
- Could have ASCII and binary interfaces and encryption,
- Needs a consistent interface for floating point numbers,
- Should be extendable,
- Open, public and widely supported,

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Literature

- Kallrath: Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis, Vieweg, 2002
- Fourer, Gay, Kernighan: AMPL, Thomson, 2003
- V.Hentenryck, Michel, Deville: Numerica, MIT Press, 1997
- Fragnière, Gondzio: Optimization Modeling Languages, 1999
- Hürlimann: LPL: A Mathematical Modeling Language — An Introduction, 2002
- Williams: Model Building in Mathematical Programming, John Wiley & Sons, 1993
- Griewank, Corliss: Automatic Differentiation of Algorithms: Theory, Implementation, and Application, SIAM, 1991

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

Acknowledgements

- I want to thank Prof. Arnold Neumaier for his help and his important advice for preparing this presentation.
- I also want to thank Dr. Oleg Shcherbina for his support and the many literature links he provided.
- Last but not least I want to thank Prof. Kallrath for inviting me to this conference and for discussing the contents of this talk.

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme

The End

Thank you for your attention

IST-2000-26063 COCONUT

Project funded by the Future and Emerging Technologies arm of the IST Programme
FET-Open scheme