A. Decision Guidance (DG): Motivation, Technical Challenges and Vision

Decision Support Systems (DSS) are widely used to support organizational and personal decision-making in diverse areas such as engineering systems, finance, business, economics and public policy. They are becoming increasingly critical with the information overload from the Internet. While the scope of DSS is broad, I view Decision Guidance Systems (DGS) as a class of DSS geared to elicit knowledge from domain experts and provide actionable recommendations to human decision-makers, with the goal of arriving at the best possible course of action. To this end, DGS may need to:

- Use and mine large amounts of data collected from multiple sources;
- Elicit knowledge about the underlying model structure from domain experts;
- Learn deterministic or stochastic models from historical data;
- Elicit metrics, performance indicators and decision objectives from decision-makers;
- Perform analytical tasks, including what-if prediction analysis and (deterministic or stochastic) optimization under diverse constraints, e.g., originating from business or engineering considerations and laws of nature;
- Present and explain actionable recommendations to decision-makers; and
- Solicit decision-makers' feedback for iterative improvement.

Until now, the practice of building DG applications has resembled the practice of developing database applications decades ago before the invention of the relational DBMS. DG applications are typically one-off and hard-wired to specific problems; require significant interdisciplinary expertise to build; are highly complex and costly; and are not extensible, modifiable, or reusable.

I believe that these deficiencies originate mainly from the diversity of the required computational tools and algorithms, each designed for a different task (such as data manipulation, predictive what-if analysis, decision optimization, statistical learning and data mining). The computational tools require the use of diverse mathematical abstractions/languages to construct input models (e.g., languages such as OPL, AMPL and GAMS for modeling mathematical programming and constraint programming problems).

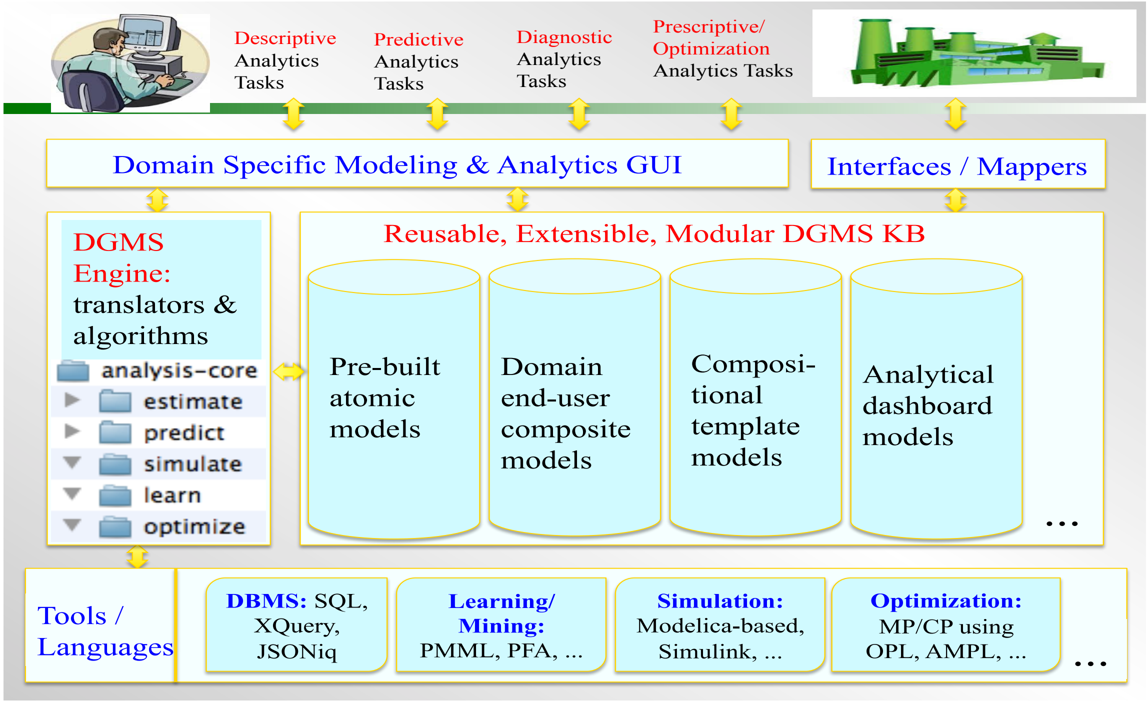
This introduces two major issues. First, the same underlying reality must often be modelled multiple times using different mathematical abstractions for different tasks/tools, instead of being modeled only once, uniformly. Second, the modeling expertise required by these abstractions/languages is typically not within the realm of DG users -- neither domain-specific users (e.g., business professionals) nor DB application and software developers (who may be used to SQL-like languages and OO programming languages such as Java). This, in turn, leads to long-duration, expensive and non-reusable development of DG applications, which must involve a team with diverse interdisciplinary expertise.

I believe that a paradigm shift for the development of DG systems is needed, as originally proposed in [79]. The key idea is to introduce and develop Decision Guidance Management Systems (DGMS), which would allow fast and easily-extensible development of DG applications, similar to easy development of DB applications using DBMS.

The conceptual architecture of DGMS is depicted in Figure 1 in the Appendix. It is centered around a reusable, modular and extensible Knowledge Base (KB) of performance models and DB views for domain–specific analysis "dashboards" (see the middle layer in Figure 1). A performance model in the KB is a formally expressed (deterministic or stochastic) computation, which describes how metrics and constraints are computed from parameters and control (decision) variables. Some performance models are *atomic,* i.e., expressed directly in the adopted language (e.g., SQL or JSONiq). Some performance models are *composite*; i.e., they are described through a graph notation showing how a model is composed of sub-models, so that domain-specific users can formulate model composition using a drag-and-drop GUI without the need to do mathematical modeling.

Declarative analysis and optimization queries can be posed against the performance models in the KB similar to DB manipulation queries against a DB. These analysis and optimization queries are then executed by the DGMS engine (see the left part of the middle layer in Figure 1). The key technical challenge lies in the development of specialized algorithms and automatic translation methods from a high-level uniform representation of models in the KB to low-level specialized models required by each of the underlying tools/algorithms, including for data manipulation, predictive what-if analysis, deterministic or stochastic decision optimization, statistical learning and data mining.

Figure 1: Conceptual Architecture of Decision Guidance Management System (DGMS)


Descriptive Analytics Tasks
Predictive Analytics Tasks
Diagnostic Analytics Tasks
Prescriptive/ Optimization Analytics Tasks
Domain Specific Modeling & Analytics GUI
Interfaces / Mappers
DGMS Engine: translators & algorithms
analysis-core
estimate
predict
simulate
learn
optimize
Reusable, Extensible, Modular DGMS KB
Pre-built atomic models
Domain end-user composite models
Composi-tional template models
Analytical dashboard models
…
Tools / Languages
DBMS: SQL, XQuery, JSONiq
Learning/ Mining: PMML, PFA, ...
Simulation: Modelica-based, Simulink, ...
Optimization: MP/CP using OPL, AMPL, ...
…