July 5th, 2019

Prescriptive Analytics for Physical Systems Models

Student: Olga Rybnytska

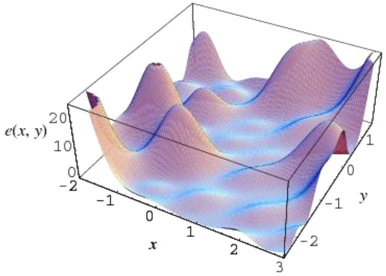
Supervisors: Torben Bach Pedersen, AAU
Robert Wrembel, PUT

AALBORG UNIVERSITY
DENMARK

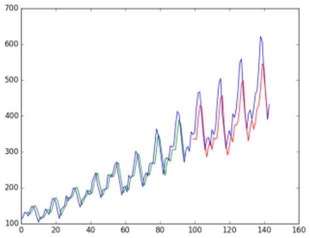
Outline

AALBORG UNIVERSITY
DENMARK

I. What is Prescriptive Analytics (PA) ?



Prescriptive Analytics - “How to make it happen?”



Predictive Analytics - “What will happen?”

Descriptive Analytics - “What happened?”

AALBORG UNIVERSITY
DENMARK

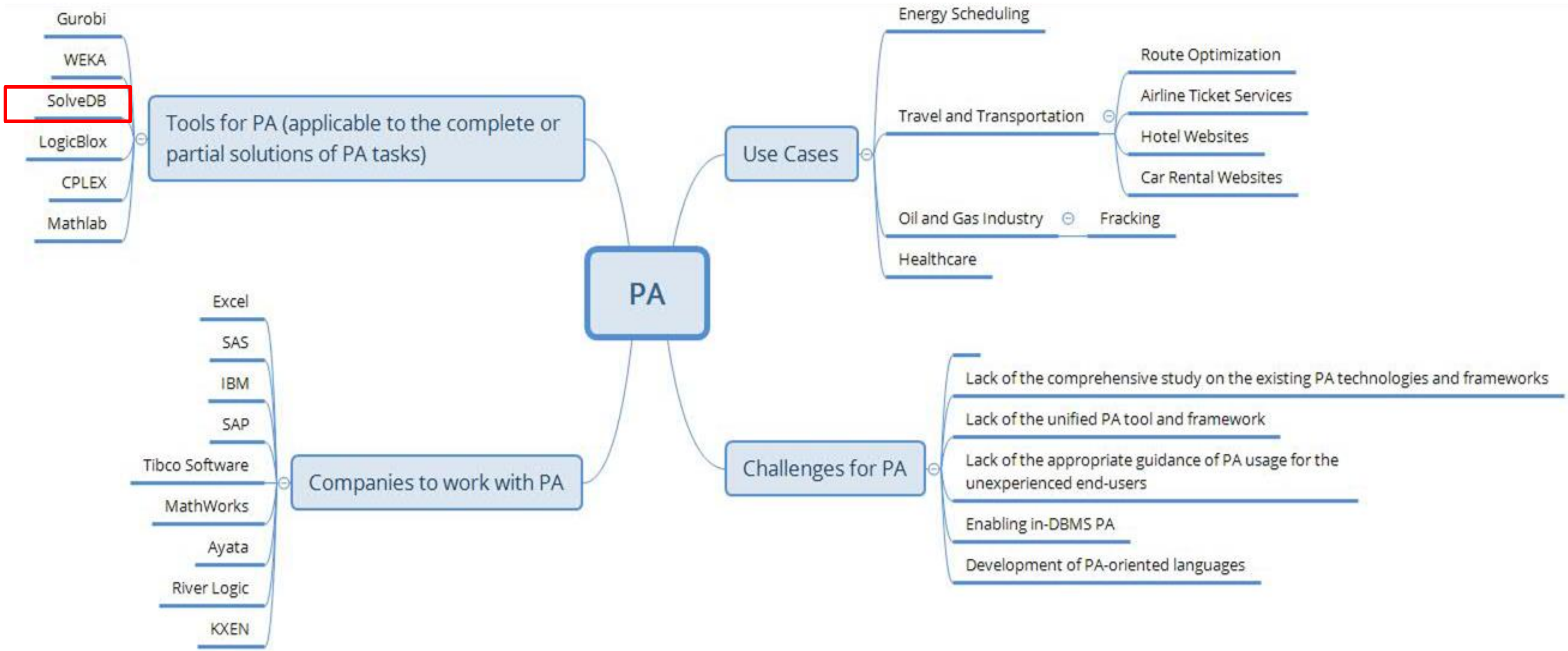
II. Motivation

Gartner Hype Cycle for Data Science [1]

- ➢ Lack of PA knowledge among data scientists
- ➢ Lack of unified PA framework
- ➢ Difficulties while switching between different types of PA tasks - system modeling, simulation, optimization
- ➢ The existing FMI-compliant simulation and optimization tools are designed for domain experts, and not traditional data analysts
- ➢ Additional skills are required from data analyst to perform the whole cycle of solving PA task

AALBORG UNIVERSITY
DENMARK

III. Background



AALBORG UNIVERSITY
DENMARK

IV. State-of-the-art

- PA term is characterized by:
  - hybrid data,
  - integrated predictions and prescriptions
  - prescriptions and side effects
  - adaptive algorithms
  - feedback mechanisms

- PA application domains:
  - InSciTe advisory (Song et al. [9]) - a PA system to facilitate the research process and provide an advice for the future
  - rBPO (Gröger et al. [10]) - recommendation-based business process optimization
  - distribution of salesforces within the company, the opportunities to increase company's profit by incorporating PA techniques into company's decision-making process (Kawas et al. [11])

AALBORG UNIVERSITY
DENMARK

IV. State-of-the-art

- No widely spread unified framework for PA applications; Soltanpoor et al. [12] suggests the prescriptive conceptual model, yet, no precise suggestions about the logical order to perform the decision-making.

- The comprehensive survey of PA software tools and frameworks was made by Frazzetto et al. [VLDBJ].

- Simulation of physical systems models - one of the most important PA tasks. Functional Mock-up Interface (FMI) [14] - a standard model representation for physical systems models simulation. Supported by Matlab [15], JModelica [16], EnergyPlus [17] and over 100 other physical systems models simulation tools.

AALBORG UNIVERSITY
DENMARK

IV. State-of-the-art

- The unified PA tool does not exist yet .
  - PostgreSQL [19] and Hadoop [18] can be used for data consolidation,
  - Matlab - for predictions,
  - JModelica - for system modeling,
  - Gurobi [20] - for optimization solutions.

- Tools to merge PA stages and support in-DBMS analytics:
  - LogicBlox [20] (consolidates predictions and optimizations),
  - Tiresias [21],
  - SolveDB [2].

    These tools handle linear programming/mixed integer programming problems; models simulation and models dynamic optimization problems are still not supported.

V. Typical PA workflow

Dynamic Optimization (DO) - an intertemporal optimization problem, where a user is to choose a sequence of actions needed to minimize/maximize the desired objective function based on a number of constraints

An example of PA task is how to control a heat pump power input to maintain the room temperature within a user-defined comfort band inside a house equipped with a heating device (e.g. a heat pump or a boiler).

Limitations: no unified tool to perform all the workflow steps.

Input Data Storage and Processing → Physical System Modeling → Prediction → Optimization → Analysis/Visualization

PostgreSQL

JModelica

R

CPLEX

WolframAlpha

Descriptive Analytics tasks

Predictive Analytics tasks

Prescriptive Analytics tasks

AALBORG UNIVERSITY
DENMARK

VI. Ph.D. Project Objectives

1. To find effective and efficient ways to store, simulate and calibrate standardized dynamic systems models within an SQL environment suitable for non-domain data analysts

1. To incorporate DO algorithms and techniques into a PA-oriented DBMS

1. To create a “wizard” that helps the inexperienced users to deal with PA tasks

1. To experimentally validate the objectives 1) - 3) based on the use cases from the energy domain, and to compare with the traditional setup

AALBORG UNIVERSITY
DENMARK

VII. Project Plan

Year	2017	2018		2019		2020
Quarter	Winter	Summer	Winter	Summer	Winter	Summer
Comprehensive literature search						
Preparation of 2-months Ph.D. study plan						
Analysis of the existing modelling systems						
Preparation of 11-months Ph.D. study plan						
Integration of the model simulation into DBMS						
Paper 3 preparation and writing						
User guide("wizard") prototype development and testing						
Paper 4 preparation and writing						
PhD thesis preparation and writing						
Milestones	MS1	MS2	MS3	MS4	MS5	MS6

Year	2017	2018	2019	2020
Quarter	Winter			Summer
User guide("wizard") prototype development and testing				
Paper 4 preparation and writing				
PhD thesis preparation and writing				
Milestones	MS1	MS2 MS3	MS4 MS5	MS6

Milestones
Submission of the Paper 4
PhD thesis submission

Milestones	MS1	MS2	MS3	MS4	MS5	MS6

Activities finished	Activities being performed	Planned activities

Table 1. Gantt chart for the Ph.D. project

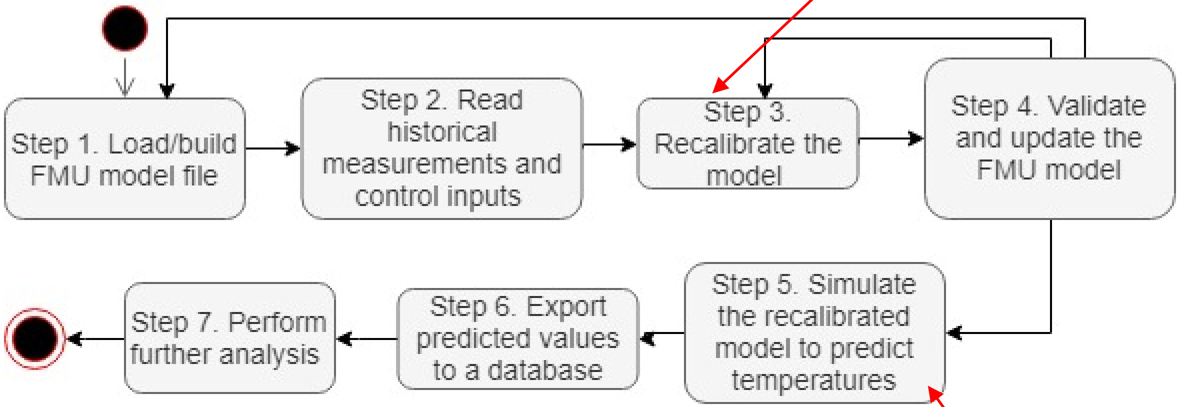
VIII. Teaching, ECTS, Papers

- 25,75 ECTS completed (86 % out of mandatory 30 ECTS)
- 772 teaching hours completed (finished teaching)
- Paper 1 “pgFMU: Integrating Data Management with Physical System Modeling” to be resubmitted to TKDE (July 2019). The next slides will present the results of this paper.

IX. pgFMU Running Example

The aim is to predict indoor temperatures inside a house heated by an electrical heat-pump (HP).

Parameters estimation - the operation of fitting model parameters to actual measurements



Running example workflow

Model simulation - the operation of calculating model outputs and states based on model inputs

IX. pgFMU Running Example

Heat pump model:

$$x(0) = x_0;$$

For $k = 1..n$:

$$x(k) = \overset{A}{\boxed{\left(1 - \frac{1}{R \cdot Cp}\right)}} \cdot x(k-1) + \overset{B}{\boxed{\left(\frac{P \cdot \eta}{Cp}\right)}} \cdot u(k) + \overset{C}{\boxed{\left(\frac{\theta_a}{R \cdot Cp}\right)}};$$

$$y(k) = \underset{E}{P} \cdot u(k);$$

Linear time-invariant (LTI) state-space model of the heat pump heated room

$Cp = 1.5kWh/°C$ is the thermal capacitance (the amount of energy needed to heat up by 1 $°C$ within 1 hour);

$R = 1.5°C/kW$ is the thermal resistance;

$P = 7.8kW$ is the rated electrical power of the heat pump;

$\eta = 2.65$ is the performance coefficient (the ratio between energy usage of the heat pump and the output heat energy);

$\theta_a = -10°C$ is the ambient temperature;

$x_0 = 21°C$ is the initial temperature;

$u(1), ..., u(n)$ are *input variables* – heat pump power rating setting in the range [0 ... 1], corresponding to [0 .. 100%] of HP power operation;

$k = 1..n$; $x(0), ..., x(n)$ are *state variables* – inside temperatures at $k = 0..n$; and

$y(1), ..., y(n)$ are *output variables* – power consumed by a heat pump at the time intervals $k = 1..n$.

IX. pgFMU Running Example

No	Operation	Step	Package	Code lines
1	Download/compile FMU model	An FMU model needs to be loaded	PyFMI	4
2	Read historical measurements and control inputs	Map the information from the sensors stored in *Measurements* table to the input, output and state variables of the FMU model	psycopg2, PyFMI, pandas	12
3	Recalibrate the model	Estimate and update model parameters to ensure better fit with the data	ModestPy, pandas	15
4	Validate and update the FMU model	Calculate the CVRMSE in order to validate the fit with the current data	PyFMI, pandas	7
5	Simulate the recalibrated model to predict temperatures	Using the updated FMU model, predict the indoor temperatures based on the known inputs	PyFMI, Assimulo, numpy	24
6	Export predicted values to a database	Insert the predicted values to the table to be updated later with the real values	psycopg2, pandas	4
7	Perform further analysis	E.g. simulate and calibrate multiple number of FMU models	psycopg2, PyFMI	22
	Total			88

Running example operations and steps

AALBORG UNIVERSITY
DENMARK

X. pgFMU Functionality

```sql
SELECT fmu_create('/home/fmus/model.fmu', 'hp1');
```

Algorithm 1: fmu_create()

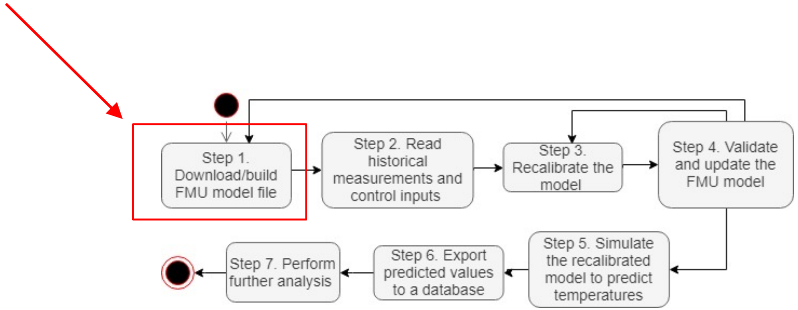
Input:

`model_id`, `fmu_file`

Output:

`fmu_model`

1: Retrieve a model identifier `model_id` specified by a user;
2: Retrieve `path` to the model;
3: Retrieve model variable names, types and values by means of internal functions `get()`, `get_model_variables()` of *PyFMI* package;
4: Based on Step 3 create an instance of *fmu_model* ;
5: Store the FMU file in a volatile memory ;



AALBORG UNIVERSITY
DENMARK

X. pgFMU Functionality

Auxiliary functions:

- fmu_variables (model_id)↦ (id, name, type, value,

```
SELECT * FROM fmu_variables('hp1') AS f WHERE
f.type = 'parameter'
```

	modelid text	modelname text	varname text	vartype text	varvalue numeric
1	hp1	Models.SISOLinearSystem	A	parameter	0
2	hp1	Models.SISOLinearSystem	B	parameter	0
3	hp1	Models.SISOLinearSystem	E	parameter	0
4	hp1	Models.SISOLinearSystem	C	parameter	0

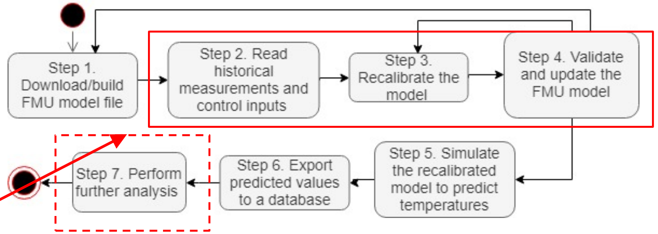
- fmu_set (model_id, name, value)

```
SELECT * FROM fmu_set('hp1', 'A', 0.56);
```

	modelid text	modelname text	varname text	vartype text	varvalue numeric
1	hp1	Models.SISOLinearSystem	B	parameter	0
2	hp1	Models.SISOLinearSystem	E	parameter	0
3	hp1	Models.SISOLinearSystem	C	parameter	0
4	hp1	Models.SISOLinearSystem	A	parameter	0.56

- fmu_get (model_id, name) ↦ value

X. pgFMU Functionality



- fmu_param_est (model_id_in, input_sql, [pars], [model_id2]) ↦ model_id_out

```sql
SELECT fmu_param_est('hp1', 'SELECT time, var_name, value
    FROM measurements', 'A, B, C, E', 'hp1')
```

ID	Model Name	Variable name	Variable Type	Initial Value	Estimated value
1	hp1	A	parameter	0.56	0.27
2	hp1	B	parameter	13.78	0.57
3	hp1	C	parameter	7.8	1.78
4	hp1	E	parameter	-4.44	-5.66

- A particular feature of the estimation of 100 heat pump model instances parameters:

```sql
SELECT * FROM generate_series(1, 100) AS id,
LATERAL fmu_param_est('hp' || id::text,
'SELECT * FROM measurements', 'A, B', 'hp' || id::text)
```

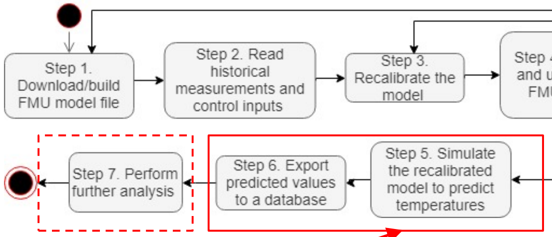
Algorithm 3: fmu_param_est()

Input:
{model_id}, {inputoutput_sql}, {pars}

Output:
{model_id_out}

1: Retrieve model_id[0] from the list of model identifiers specified by a user;
2: Based on the inputoutput_sql[0], retrieve the *measurements* table;
3: Retrieve the input variables values from the *measurements* table ;
4: Retrieve the parameters to be estimated from pars;
5: For model_id[0], run Global Search and Local Search estimation algorithms;
6: Store estimation results to *Parameters* table;
7: Update all model instances from {model_id} with the new parameter values;
8: For all remaining models (model_id[1], ... model_id[n], n = length({model_id})) run only Local Search algorithm, using estimation results from Step 6 as initial guess;
9: Store estimation results for model_id[1], ... model_id[n] to *Parameters* table;
10: Store the new model instances as {model_id_out}

AALBORG UNIVERSITY
DENMARK



X. pgFMU Functionality

- fmu_simulate (model_id, input_sql, [time_from], [time_to]) ↦ (time, var_name, sim_value, real_value)

```sql
SELECT time, var_name, sim_value, real_value
FROM fmu_simulate('hp1', SELECT time, var_name, value
FROM measurements WHERE var_name IN ('''u''')') AS f
```

Time	Model Name	State var	State var real	State var simulated
0	hp1	x	20.7507	20.188
1	hp1	x	23.6231	19.998
2	hp1	x	20.543	19.808
...	...		...	...

- A particular feature of the simulation of 100 heat pump model instances:

```sql
SELECT * FROM generate_series(1, 100) AS id,
LATERAL fmu_simulate('hp' || id::text ,
'SELECT * FROM measurements') AS f
```

Algorithm 2: fmu_simulate()

Input:

`model_id`, `SQL query`

Output:

ModelSimulation table.

1: Retrieve a model identifier `model_id` specified by a user;
2: Retrieve the FMU model instance based on `model_id`;
3: Retrieve the *Measurements* table;
4: Map the input (u), output (y) and state (x) variables values in the *measurements* table to the input (`HP_range`), output (`HP_power`) and state (`Indoor_Temp`) variables of the FMU model instance ;
5: Perform model initialization by simulating a model from the initial time t_0 to initial time + ε (given ε is a linear interpolation between time values t_0 and t_1);
6: Perform *model_simulation* using PyFMI Python package with `simulate()` function;
7: Store simulation results in *ModelSimulation* table;

AALBORG UNIVERSITY
DENMARK

XI. pgFMU Experimental Evaluation

Model HP0 - modification of the running example, heat pump model with no inputs (heat pump power is kept at a constant rate)

Model ID	Measurements dataset	Inputs	Outputs	Parameters
HP0	NIST Engineering	No inputs	– heat pump power con-	– thermal capacitance *Cp*
HP1	NIST Engineering Lab's Net Zero Energy	– heat pump power rat-	– heat pump power con-	– thermal capacitance *Cp*,
Class-room	Data from one of the class-rooms in the test facility [35]	– solar radiation *solrad*, – outdoor temperature *tout*, – number of occupants *occ*, – damper position *dpos*, – radiation valve position *vpos*.	– indoor temperature *t* (state variable)	– solar heat gain coefficient *shgc*, – zone thermal mass factor *tmass*, – external wall thermal resistance *RExt*, – occupant heat generation effectiveness *occheff*.

FMU models to test pgFMU functionality upon

Model HP1 - running example

Model Classroom - a thermal network model [23] represented by a classroom (139 m2) in a 8500 m2 teaching university building OU44 at the SDU Campus Odense (Odense, DK)

XI. pgFMU Experimental Evaluation

1. Configurations:
   a. C_Python - using Python IDE and Python packages functionality.
   b. C_pgFMU - using pgFMU functionality with no multi-model optimization features activated.
   c. C_pgFMU+ - using pgFMU functionality with multi-model optimization features activated.
2. Scenarios:
   a. Single model scenario - C_Python, C_pgFMU, and C_pgFMU+.
   b. Multi-model scenario - C_Python, C_pgFMU, and C_pgFMU+.

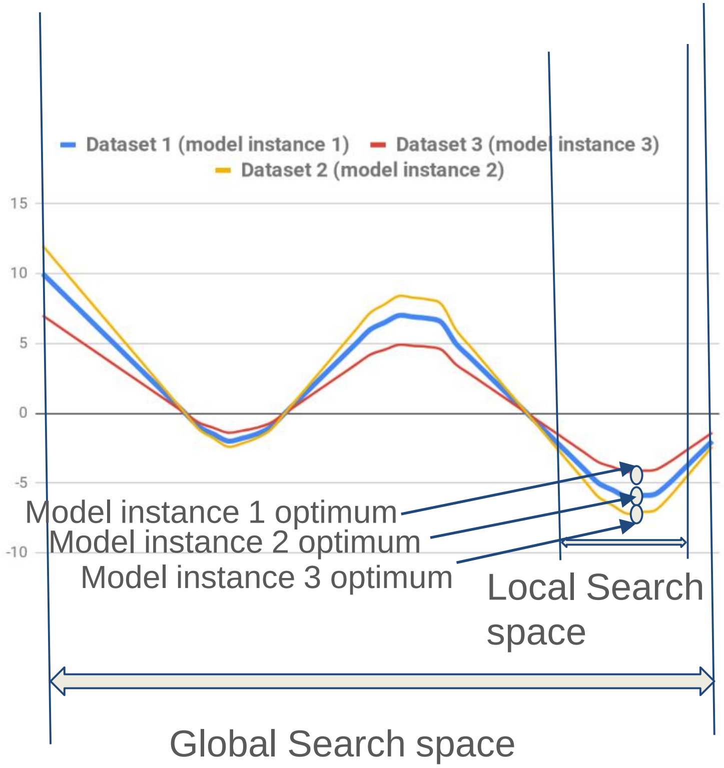
3. Assessment: Model Quality, Performance, Usability.

AALBORG UNIVERSITY
DENMARK

XI pgFMU Experimental Evaluation

pgFMU+ optimization: for estimating parameters of the model, reduce the search space by using “initial guess” technique.
This technique is applicable for multiple model instances.

Main idea: after running Global Search + Local Search algorithms for one model instance, reuse the results for the remaining model instances, assuming the model instances are of the same structure and input time series have the same distribution.



AALBORG UNIVERSITY
DENMARK

XI. pgFMU Experimental Evaluation

Model Quality, single model scenario

	C_Python		C_pgFMU(+)	
	Param. values	RMSE	Param. values	RMSE
HP0	Cp: 84	0.7701	Cp: 84	0.7702
	R: 0.017		R: 0.0017	
HP1	Cp: 86	0.5445	Cp: 86	0.5445
	R: 0.009		R: 0.009	
Classroom	RExt: 4	1.6445	RExt: 4	1.6442
	occheff: 1.478		occheff: 1.478	
	shgc: 3.246		shgc: 3.246	
	tmass: 50		tmass: 50	

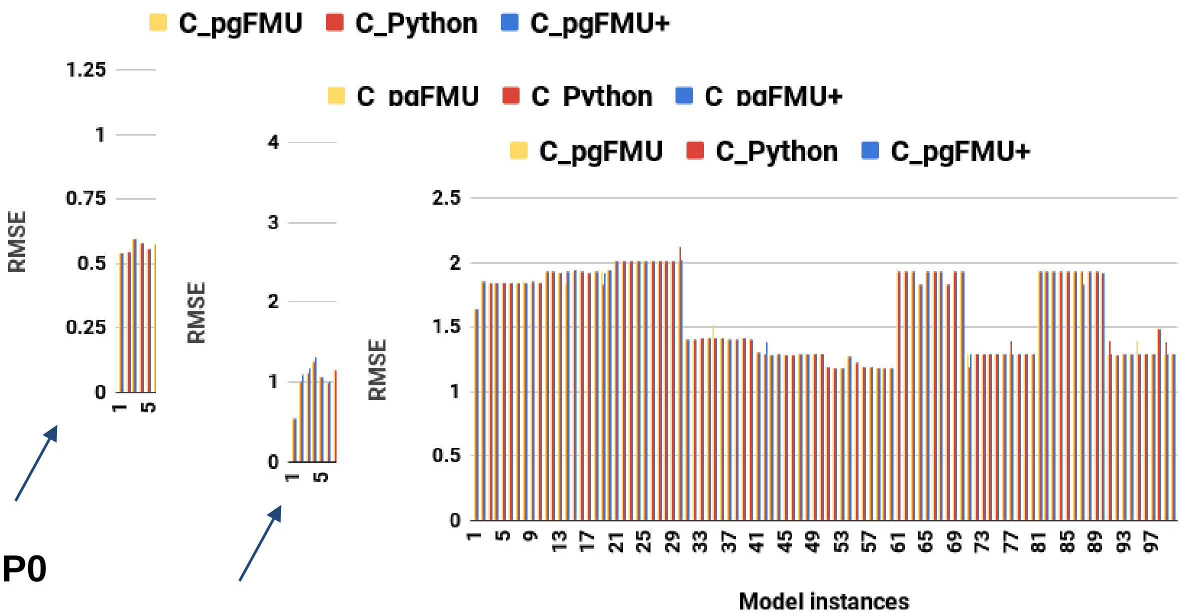
Model Quality comparison within Python and pgFMU, pgFMU+ configurations

AALBORG UNIVERSITY
DENMARK

XI. pgFMU Experimental Evaluation

Model Quality, multi model scenario

RMSE comparison, 100 model instances



HP0

HP1

Classroom

AALBORG UNIVERSITY
DENMARK

XI. pgFMU Experimental Evaluation

Performance, single model scenario

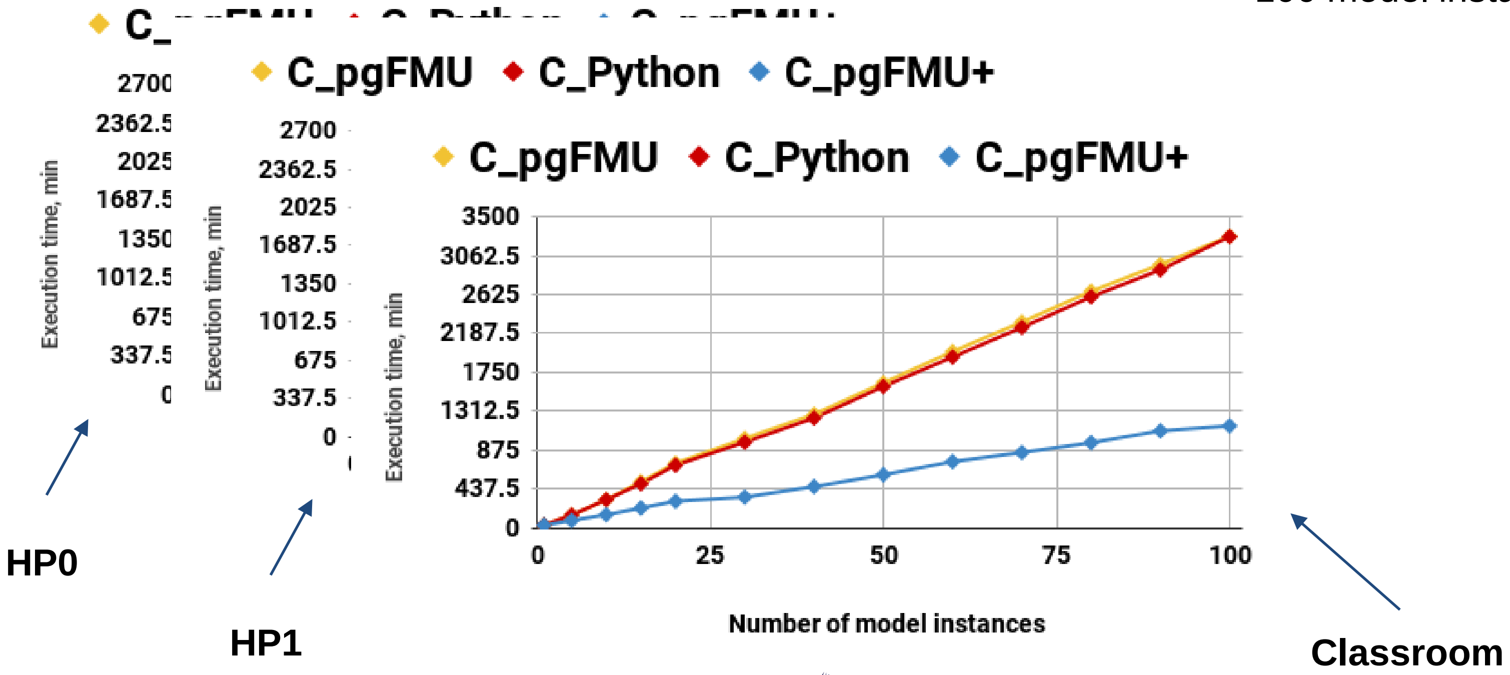
ID	Operation	execution time, s						lines of code	
		HP0		HP1		Classroom		-	-
		C_Python	C_pgFMU(+)	C_Python	C_pgFMU(+)	C_Python	C_pgFMU(+)	C_Python	C_pgFMU
1	Load FMU	0.02	0.025	0.02	0.021	0.03	0.03	4	1
2	Read historical measurements & control inputs	0.02	0.021	0.03	0.031	0.04	0.041	12	-
3	Calibrate the model	1262.99	1264.18	1972.68	1970.88	1682.2	1680.16	15	1
4	Validate and update FMU model	0.01	-	0.01	-	0.01	-	7	-
5	Simulate FMU model	0.16	0.214	0.2	0.22	0.35	0.44	24	1
6	Export predicted values to a DBMS	0.06	-	0.06	-	0.05	-	4	-
	Total	**1263.26**	**1264.44**	**1973**	**1971.15**	**1682.68**	**1680.66**	**66**	**3**

Configurations comparison, 1 model instance

XI. pgFMU Experimental Evaluation

Performance, multi model scenario

Workflow execution time comparison, 100 model instances



AALBORG UNIVERSITY
DENMARK

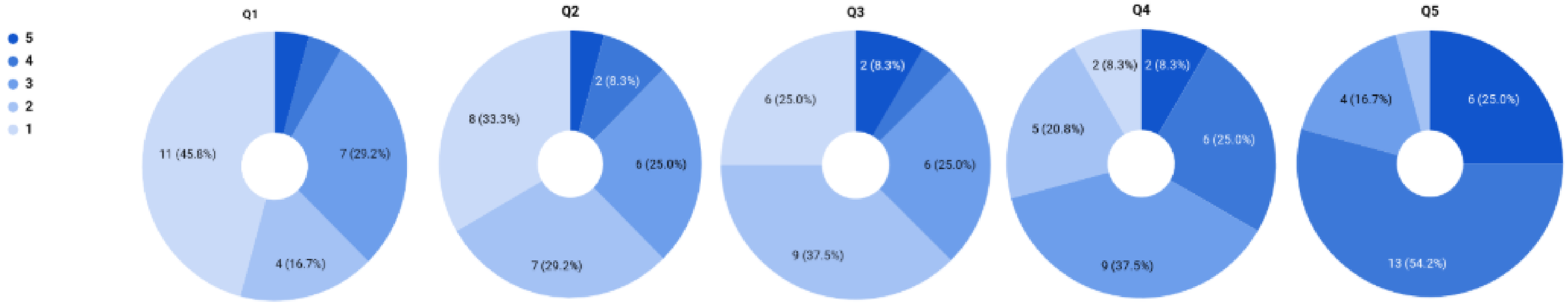
XI. pgFMU Experimental Evaluation

Usability

User testing session with 24 master students from Poznan University of Technology (PUT).

Pre-assessment questionnaire (1 - very little, 5 - very much):

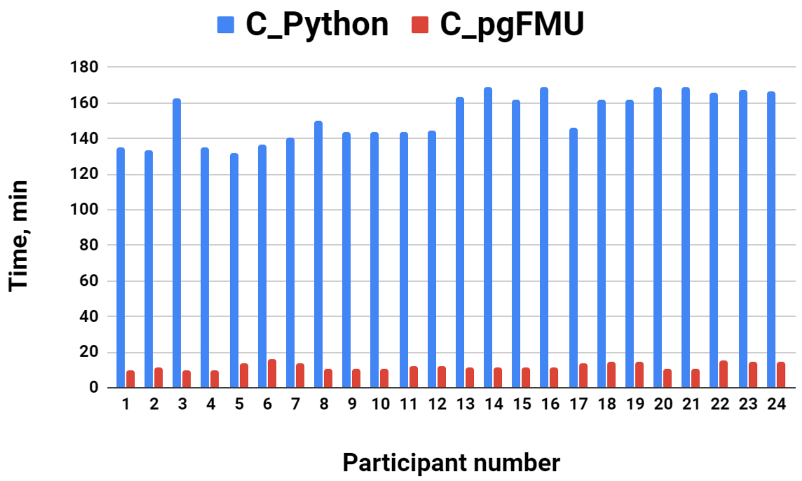
Q1. How can you estimate your knowledge in energy systems and physical systems modeling?
Q2. How familiar are you with model simulation and model calibration process?
Q3. How familiar are you with model simulation software(s)?
Q4. How comfortable are you with using Python IDE?
Q5. How comfortable are you with using SQL?



Pre-assessment questionnaire results

XI. pgFMU Experimental Evaluation

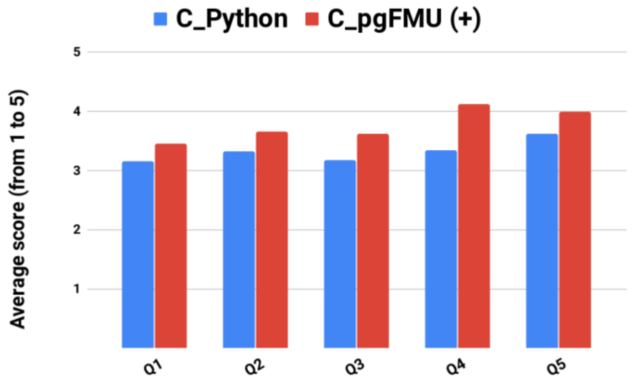
Usability



Student time per workflow, min (excluding UDFs runtime)

Post-assessment questionnaire:

Q1. How was it to retrieve information about model variables?
Q2. How was it to set model parameters?
Q3. How was it to calibrate the model?
Q4. How was it to simulate the model ?
Q5. Overall satisfaction with configuration functionality.



1 - very difficult/unsatisfactory, 5 - very easy/satisfactory

Post-assessment questionnaire results

AALBORG UNIVERSITY
DENMARK

XI. pgFMU Experimental Evaluation

Usability

Strong points of Python configuration	Strong points of pgFMU configuration
• “Better to debug, analyze” • “More functionality” • “Control over program flow” • “Data visualization option”	• "Data and model in one place" • "Easy to run and understand" • "Simplicity, no need to use or import external tools" • "Familiar SQL syntax"
Weak points of Python configuration	**Weak points of pgFMU configuration**
• "A lot of unknown new modules and packages" • "No one ready package to do everything" • "A lot of code to set up configuration, some functions not intuitive" • "You need practise[sic] to understand" • "Too much control makes it harder"	• "Not so much configuring available" • "I don’t see any significant besides maybe installation of the package on postgres[sic]" • "Specific database implementation"

Participants opinion about both configurations

XII. pgFMU Conclusions and Future Work

- pgFMU - the first DBMS extension to support simulation, calibration and validation of physical systems dynamic models within a single DBMS environment.
- pgFMU provides time-efficient functionality to store, simulate, calibrate and analyze an arbitrary number of FMU models.
- On average 2.9 times execution time gain in comparison to the traditional workflow, and 22 times less code lines.
- pgFMU is up to 12.5 times faster in terms of development time for the arbitrary user-defined workflow.

AALBORG UNIVERSITY
DENMARK

XIII. Selected PA-supporting Tools Comparison

Typical PA Workflow

	Input Data Storage and Processing	Physical System Modeling	Predic-tion	Optimiza-tion	Analysis/ Visualization
SolveDB	+		+	+/-	+/-
pgFMU	+	+	+/-		+/-
Matlab		+/-	+	+	+
JModelica		+	+/-	+	+/-

XIV. Next Steps towards Unified PA Tool Creation

- SolveDB [2] - a Postgres-based DBMS with the native support for in-DBMS optimization, constraint satisfaction and domain-specific problems;
- Provides a set of built-in solvers for Linear Programming (LP)/Mixed Integer Programming (MIP), Global Optimization (GO);
- Integrates solver into DBMS backend, therefore, making database-based problem solutions more easy and user-friendly.

Maximize	$0.6x_1 + 0.5x_2$
Subject To	$x_1 + 2x_2 \leq 1$

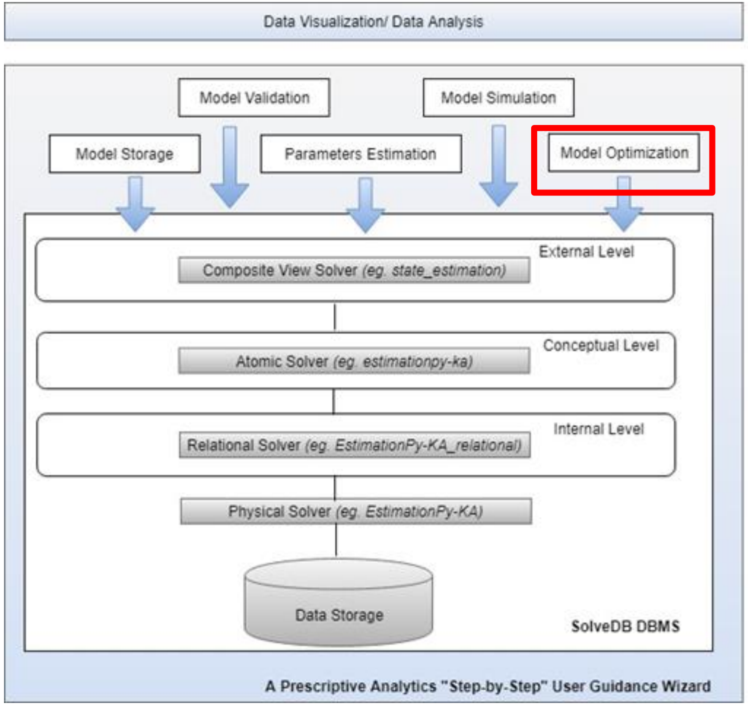
Example of a simple
linear optimization problem

A view solver produces a solution based on the model instance descriptor and parameter-value pairs

```
SOLVESELECT x1, x2 IN (SELECT x1, x2 FROM data) AS u
MAXIMIZE   (SELECT 0.6*x1 + 0.5*x2 FROM u)
SUBJECTTO  (SELECT x1+2*x2 <= 1 FROM u),
           (SELECT 3*x1+x2 <= 2 FROM u)
USING solverlp();
```

Query example

XIV. Next Steps towards Unified PA Tool Creation



Extended SolveDB+ Architecture

Composite view solver

```
SOLVESELECT x1, x2 IN (SELECT x1, x2 FROM data) AS u
MAXIMIZE  (SELECT 0.6*x1 + 0.5*x2 FROM u)
SUBJECTTO (SELECT x1+2*x2 <= 1 FROM u),
          (SELECT 3*x1+x2 <= 2 FROM u)
USING solver_do (model_id)
```

Interpreter (e.g. Optimica language for DO problems)

Non-linear physical solver (e.g. IPopt)

XIV. Next Steps towards Unified PA Tool Creation

Paper 2 “Bringing Model Dynamic Optimization into Prescriptive Analytics DBMSes” (ICDE, October 2019)

- To continue the integration of the DO techniques into DBMS.
- To enable automatic DO solver generation.
- To build model optimization on top of the upgraded pgFMu extension from Paper 1.
- To provide a native support for the DO solvers.
- Focus mainly on enabling the in-DBMS optimal control methods and techniques.

XIV. Next Steps towards Unified PA Tool Creation

Paper 3 “Data-Driven State-Based Simulation and Calibration of Residential Heat Pump Models” (e-energy, January 2020)

- The real-world example of physical system modelling located in Switzerland to be considered.
- A number of houses are equipped with the heat pump and boiler physical devices.
- The real-time data from sensors (heat pump power meter, boiler power meter, boiler temperature sensor, room temperature sensor) to be fetched into the model.
- The boiler and heat pump model simulation, state and parameters estimation, and model optimization will be done via the usage of pgFMU DBMS extension from Paper 1 and 2.
- The system predicts the state of the residential heat pump and boiler, its future energy demand and the room temperature.

XIV. Next Steps towards Unified PA Tool Creation

Paper 4 “A Unified Prescriptive Analytics Tool” (TODS/VLDB, August 2020)

- A journal paper
- DO-DBMS platform is enhanced by “wizard” integration.
- “Wizard” will guide users through the PA task solution process by bringing the step-by-step guidance:
  - helping the user to choose the appropriate solver/function for a specific type of PA task, and
  - navigating through the PA stages (processing the measurements for the system modeling, model prediction, model simulation, state and parameters estimation of the model, and model optimization).
- Running example - Swiss case (Paper 3) and thermal energy consumption prediction (collaboration with PUT).

AALBORG UNIVERSITY
DENMARK

XV. Intended Project Contribution/ Improvement over the State-of-the-art

- Improving the way of non-domain data analysts interaction with PA tasks by designing a framework for in-DBMS cyber-physical models storage, simulation, and calibration.
- Addressing the issue of poor direct database inputs support from the existing simulation and optimization software;
- Enabling in-DBMS model optimization functionality for different types of PA tasks;
- Providing user support by creation of a “wizard” - a step-by-step guidance tool to smooth the process of PA task solving;
- Refining the practises of residential heat pumps models storage, simulation, calibration and optimization.

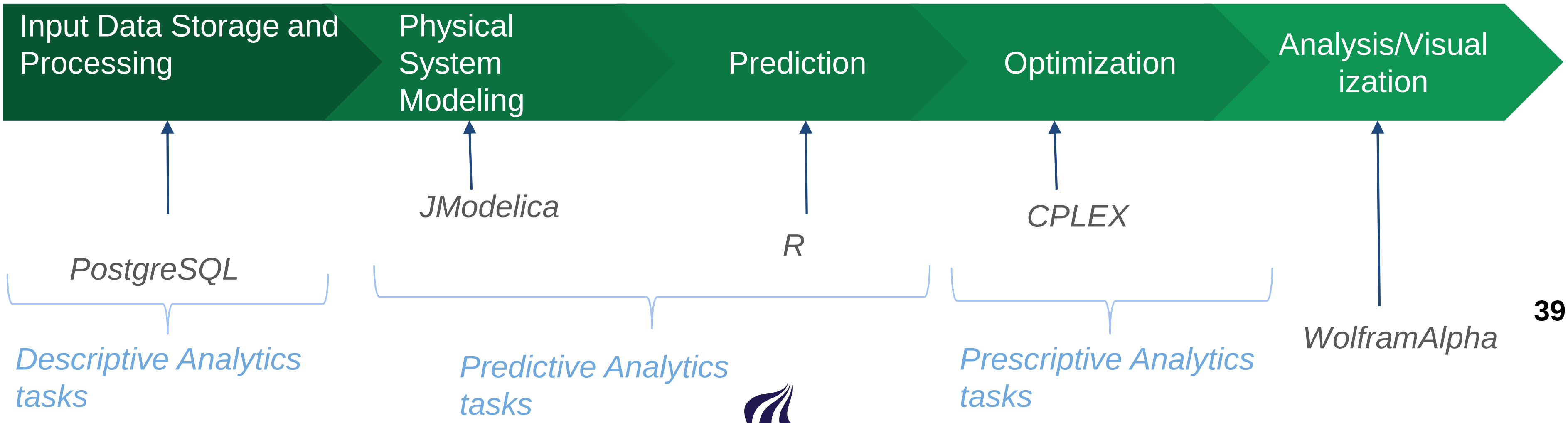
AALBORG UNIVERSITY
DENMARK

References

1. *Technology Research | Gartner Inc.*. (2017). *Gartner.com*. Retrieved 05 December 2017, from https://www.gartner.com/technology/home.jsp
2. Šikšnys, L., & Pedersen, T. B. (2016, July). Solvedb: Integrating optimization problem solvers into sql databases. *In Proceedings of the 28th International Conference on Scientific and Statistical Database Management* (p. 14). ACM.
3. Healy, William, A. Hunter Fanney, Brian Dougherty, W. Vance Payne, Tania Ullah, Lisa Ng, and Farhad Omar. "Net Zero Energy Residential Test Facility Instrumented Data; Year 2," January 2017. https://doi.org/10.18434/T46W2X
4. SAP Software Solutions | Business Applications and Technology. (2019). SAP. Retrieved 17 March 2019, from https://www.sap.com/index.html
5. IBM - Danmark. (2019). Ibm.com. Retrieved 17 March 2019, from https://www.ibm.com/dk-da/
6. Global Leader in Integration and Analytics Software. (2019). TIBCO Software Inc.. Retrieved 17 March 2019, from https://www.tibco.com/
7. Prescriptive Planning and Performance Management | River Logic. (2019). RiverLogic.com. Retrieved 17 March 2019, from https://www.riverlogic.com/
8. Song, S. K., Kim, D. J., Hwang, M., Kim, J., Jeong, D. H., Lee, S., ... Sung, W. (2013, December). Prescriptive analytics system for improving research power. In Computational Science and Engineering (CSE), 2013 IEEE 16th International Conference on (pp. 1144-1145). IEEE
9. Gröger, C., Schwarz, H., Mitschang, B. (2014, May). Prescriptive analytics for recommendation-based business process optimization. In International Conference on Business Information Systems (pp. 25-37). Springer, Cham.
10. Kawas, B., Squillante, M. S., Subramanian, D., Varshney, K. R. (2013, December). Prescriptive analytics for allocating sales teams to opportunities. In Data Mining Workshops (ICDMW), 2013 IEEE 13th International Conference on (pp. 211-218). IEE
11. Soltanpoor R., Sellis T. (2016) Prescriptive Analytics for Big Data. In: Cheema M., Zhang W., Chang L. (eds) Databases Theory and Applications. ADC 2016. Lecture Notes in Computer Science, vol 9877. Springer, Cham
12. Laurynas Siksnys. (2015). Towards Prescriptive Analytics in Cyber-Physical Systems. Ph.D. Dissertation. Dresden University of Technology
13. Functional Mock-up Interface. (2018). Fmi-standard.org. Retrieved 6 September 2018, fromhttps://fmi-standard.org/
14. MATLAB - MathWorks. (2019). Mathworks.com. Retrieved 17 March 2019, from https://www.mathworks.com/products/matlab.html
15. Åkesson, J., Gäfvert, M., & Tummescheit, H. (2009, February). Jmodelica—an open source platform for optimization of modelica models. In Proceedings of MATHMOD
16. EnergyPlus | EnergyPlus. (2019). Energyplus.net. Retrieved 17 March 2019, from https://energyplus.net/
17. Apache Hadoop. (2019). Hadoop.apache.org. Retrieved 17 March 2019, fromhttps://hadoop.apache.org
18. Momjian, B. (2001). PostgreSQL: introduction and concepts (Vol. 192). New York: Addison-Wesley
19. Gurobi Optimization - The State-of-the-Art Mathematical Programming Solver. (2019).Gurobi.com. Retrieved 17 March 2019, from http://www.gurobi.com
20. Aref, M., ten Cate, B., Green, T. J., Kimelfeld, B., Olteanu, D., Pasalic, E., ... & Washburn, G. (2015, May). Design and implementation of the LogicBlox system. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (pp. 1371-1382). AC
21. Meliou, A., & Suciu, D. (2012, May). Tiresias: the database oracle for how-to queries. In Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data (pp. 337-348). ACM
22. K. Arendt, C.T. Veje (2019). MShoot an Open Source Framework for Multiple Shooting MPC in Buildings. 16th IBPSA International Conference and Exhibition Building Simulation 2019, Rome 2-4 September, 2019.

Questions?

Typical PA workflow



AALBORG UNIVERSITY
DENMARK

39