A Comparison of the Performance of a Mathematic Expression Parser in Heat Load Modelling

Erik Král, Petr Čápek

***Abstract*—** This article presents a comparison of the performance of a Mathematical Expression Parser in Heat Load Modelling for a District Heating System. Many simulation problems are connected to dynamic function compositions. We have developed a simulation tool for heat load prediction with a user interface build as an excel add-in. A composition pattern and a mathematic math expression parser are compared in this article.

***Keywords*—** District Heating, Heat Load, Modelling, Composite pattern, Math Parser, Dynamic Compilation.

I. Introduction

THIS paper compares the performance of a composite pattern, a math parser using dynamic compilation. The problem is demonstrated in the case study of an Excel add-in application for Heat load modelling [1].

The automatic recognition of mathematic expressions is a complex problem that consists of two major stages, symbol recognition and structural analysis [2][3]. This article only deals with mathematical expressions in text format and its structural analysis and its dynamic compilation in .NET framework.

The main aim is to compare the performance of parsed and dynamically compiled expressions to a static complied expression using a classic composite pattern.

II. Problem Formulation

District heating networks have dynamic properties such as water flow and the propagation of heat from a combined heat and power plant to consumers and back again. Such networks can be mathematically modelled. Some of these methods are computationally intensive and require full physical modelling of the networks but there are also simplified methods.

Typical methods for heat load modelling are Time-series analyses, such as Autoregressive Moving Average with Extraneous Input, modelling the physical structure and the behavior of the system and neural networks [4][5][6][7][8][9].

Heller [6] in his work presents these approaches, a top-down approach which describes the individual sink by an analysis of heat load data for a whole system or plant and a bottom-up approach which attempts to estimate the total heat load by describing the individual sink. Werner [7] presents a model in which heat load is modelled as a sum of an independent model variable element (e.g. external temperature, wind speed) and corresponding coefficients that adjust the element. These elements can be grouped into load components [7]:

Heller also estimated component significances as you can see in Table 1.

Table 1: Heat load components significance [6]

Load Component	Significance estimations in %
Space heating for buildings	60
Domestic hot water preparation.	30
Distribution loss	6-8
Additional workday loads	4-2

Dotzauer [10], in contrast to Heller's components, presents a simple model that consists of a sum of the temperature dependent component and the remaining part (mostly dependent on time).

We have developed a model similar to the Dotzauer, wherein heat load consists of a sum of the temperature dependent and the remaining part, mostly dependent on time.

There are two variants of a total heat load model, the additive and the multiplicative model. The former uses a sum of two components and the latter uses a components product. In some cases, the multiplicative model provides more precise results than the additive [1]. For the purpose of heat load modelling, a simple model can be used. The district heating system can be approximated by the load centre of the mass of the system [11]:

$$P(t) = \dot{m}(t)c\left(\vartheta_1\left(t - \frac{T_{D1}}{2}\right) - \vartheta_0\left(t + \frac{T_{D0}}{2}\right)\right) \quad (1)$$

where

- $P(t)$ represents the heat load,
- $\dot{m}(t)$ represents the measured mass flow,
- c represents the specific heat capacity,
- ϑ_1 represents supply temperature,
- ϑ_0 represents return temperature,
- T_{D1} represents the supply line transport time,
- T_{D0} represents the transport time of return line,
- t represents time.

We can simplify the model so that the transport time is only dependent on the mass flow and the total mass volume of a

All authors are with Faculty of Applied Informatics, Tomas Bata University in Zlín, Nad Stráněmi 4511, 760 05 Zlín (phone: +420-57-603-5188; fax: +420 57-603-2716; e-mail: ekral@ fai.utb.cz

district heating network. Transport times can be calculated based on [11]:

$$R = \int_{t-T_{D1}}^{t} \dot{m}(\tau)\, d\tau \quad (2)$$

$$R = \int_{t}^{t+T_{D0}} \dot{m}(\tau)\, d\tau \quad (3)$$

where

R	represents the known mass volume,
T_{D1}	represents the unknown transport time of the supply line,
T_{D0}	represents the unknown transport time of the return line.

The heat load is approximated by the sum of time dependent and temperature dependent components:

$$f_P(t, \vartheta_{ex}) = f_{time}(t) + f_{temp}(\vartheta_{ex}) \quad (4)$$

where

$f_{time}(t)$	represents the time dependent, component,
ϑ_{ex}	represents the outdoor temperature,
$f_{temp}(\vartheta_{ex})$	represents the outdoor temperature, dependent component.

A. Temperature Dependent Component

The temperature dependent component is at certain periods inversely proportional to the external temperature and we have chosen the Piecewise linear function, Unified-Richards function defined as [12]:

$$f_{URM}(\tau) = \left(1 + (d-1)e^{\frac{-K(\tau - T_i)}{d^{\frac{d}{(1-d)}}}}\right)^{\frac{1}{(1-d)}} \quad (5)$$

where

A	is upper asymptote,
K	is slope at inflection,
T_i	is time at inflection,
$d^{\frac{d}{(1-d)}}$	is proportion of upper asymptote at inflection.

And fourth degree polynomial:

$$f_{P4}(\vartheta_{ex}) = a_0 + a_1\vartheta_{ex} + a_2{\vartheta_{ex}}^2 + a_3{\vartheta_{ex}}^3 + a_4{\vartheta_{ex}}^4 \quad (6)$$

B. Time Dependent Component

The daily heat load pattern is typified by its morning and evening peaks. Thus, the time dependent component is approximated by the sum of the two peak functions. The Hourly coefficients and Hybrid of Gaussian and truncated exponential function (EGH) were selected as most the convenient functions. The EGH function has the capability to incorporate asymmetric peaks and its fast convergence [13]. The Hybrid of Gaussian and truncated exponential function is defined as:

$$f_{EGH}(\tau) = \begin{cases} H\, exp\left(\frac{-(h-h_m)^2}{2\sigma^2 + \tau(h-h_m)}\right), & 2\sigma^2 + \tau(h-h_m) > 0 \\ 0, & 2\sigma^2 + \tau(h-h_m) \le 0 \end{cases} \quad (7)$$

And the $f_{time}(t)$ function is then the sum of two EGH functions. Because the function is periodical and describes a 24hour daily pattern we have to shift time by using a time offset and modulo function to match the daily minimum around 1 am.

There are two variants of model approximation, the combination of the Gaussian and truncated exponential function and Unified-Richards:

$$f_1(\tau, \vartheta_{ex}) = f_{EGH}(\tau) + f_{EGH}(\tau) + f_{URM}(\vartheta_{ex}) \quad (8)$$

And the combination of the Gaussian and truncated exponential function and Fourth degree polynomial:

$$f_2(\tau, \vartheta_{ex}) = f_{EGH}(\tau) + f_{EGH}(\tau) + f_{P4}(\vartheta_{ex}) \quad (9)$$

C. Particle Swarm Algorithm

The Particle swarm algorithm (PSO) [14][15] was chosen as the numeric optimisation algorithm suitable for problems without the explicit knowledge of the gradient of the function to be optimised. Traditional PSO (TPSO), should be written in this form:

$$V_{id}(k+1) = \omega V_{id}(k) + c_1 r_1\left(PB_{id}(k) - X_{id}(k)\right) + c_2 r_2\left(GB_d(k) - X_{id}(k)\right) \quad (10)$$

$$X_{id}(k+1) = X_{id}(k) + V_{id}(k) \quad (11)$$

where

i	represents the particle index i = 1,2, … NP,
NP	represents the number of particles in swarm,
d	represents the dimension index $d = 1,2, \dots D$,
D	represents the dimension of the solution space,
k	represent the index of iteration,
$X_{id}(k)$	represents the particle position,
$V_{id}(k)$	represents particle velocity
$PB_{id}(k)$	represents the particle best position,
$GP_d(k)$	represents the swarm best position,
ω	represents the inertia component,
c_1	represents the social component,
c_2	represents the cognitive component,
r_1, r_2	are uniformly distributed random numbers in interval $[0, 1]$.

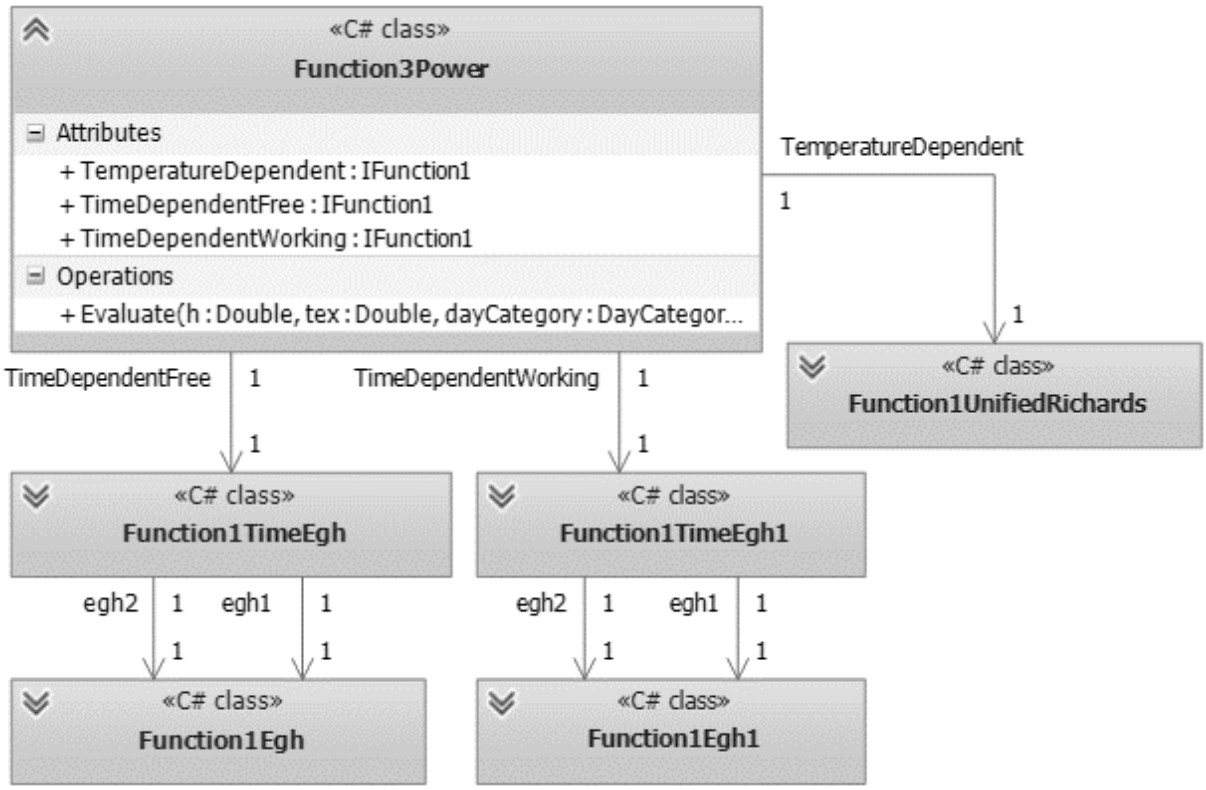

Fig. 1: Function class diagram

The particle velocity is limited to $V_{id}(k) \in [-V_{max}, V_{max}]$, where V_{max} is the maximum particle velocity. The number of particles *NP* is usually set at two times more than the dimension D. The inertia component ω is set at about 0.8, the social component c_1 is set at about 1.4 and the cognitive component c_2 is set at about 0.6. We use *MaxDistQuick* as a stopping criterion as described in [16]. The optimization is stopped when the maximum distance of the majority of the particles is below the threshold *eps* or the maximum number of iteration is reached.

III. Excel Add-in

We developed an excel add-in for a user friendly prediction of heat load wherein a user can independently select:

- Additive or multiplicative variant
- Temperature dependent function
- Time dependent function

Users can select these temperature dependent functions [1]:

- Unified-Richards model [11]
- Fourth degree polynomial
- Piecewise linear function

And these time dependent functions:

- Gaussian and truncated exponential function [12]
- Hourly coefficients

In total there can be 18 combinations of functions and in the future a user may want use new functions. These functions are quite complex and can consume a lot of CPU power. The model parameters are estimated using the Standard Particle Swarm Optimization [13]. The predictions are measured using the data from the week following the identification period. The models are compared using Mean Absolute Percentage Error (MAPE).

We used a composite-like pattern for heat load function representation as depicted in Figure 1 and 3. Each component is represented by reference to interface *IFunction1*, which represents the function of one parameter. The class *Function3Power* describes the function of three parameters time, function and day category and has three fields:

- *TemperatureDepended*
- *TimeDependentFree*
- *TimeDependentWorking*

And each of these field types of interface *IFunction1* and concrete implementation is injected in constructor.

The problem is that the function must be statically compiled as a .NET assembly and installed on a user's computer. Our aim is to use a mathematic expression parser and a compiler so users can input any function at the run time without the need for changes in source code. In addition, the mathematic parser could decrease the overall performance.

IV. Problem Solution

We measured five functions and function combinations:

1. Unified-Richards function.
2. Hybrid of Gaussian and truncated exponential function.
3. Fourth degree polynomial.
4. Combination of the Gaussian and truncated exponential function and Unified-Richards.
5. Combinations of the Gaussian and truncated exponential function and Fourth degree polynomial.

The Dynamic Expresso [17] tool was selected after the performance comparison [2] as the standard expression interpreter with good performance. Dynamic Expresso is an expression interpreter for C#. It interprets C# statements by converting it into .NET delegates that can be invoked as the standard delegate [17].

We ran each function 1000 times and computed the average time of execution. Firstly, we tested the function as the composition of a compiled function and later using a delegate generated by the Dynamic expresso.

The testing machine has the Intel i7 processor. The results are shown in Table 2. As you can see in Table 1, the expression interpreter provides the same or in some cases even better results than the composition of the native function in Figure 1.

Table 2: Measurement results in milliseconds

Id	Function	Composition	Expresso
1	$f_{URM}(\tau)$	1.295	1.272
2	$f_{P4}(\vartheta_{ex})$	1.325	0.994
3	$f_{EGH}(\tau)$	1.229	0.456
4	$f_1(\tau, \vartheta_{ex})$	2.238	1.464
5	$f_2(\tau, \vartheta_{ex})$	2.282	1.231

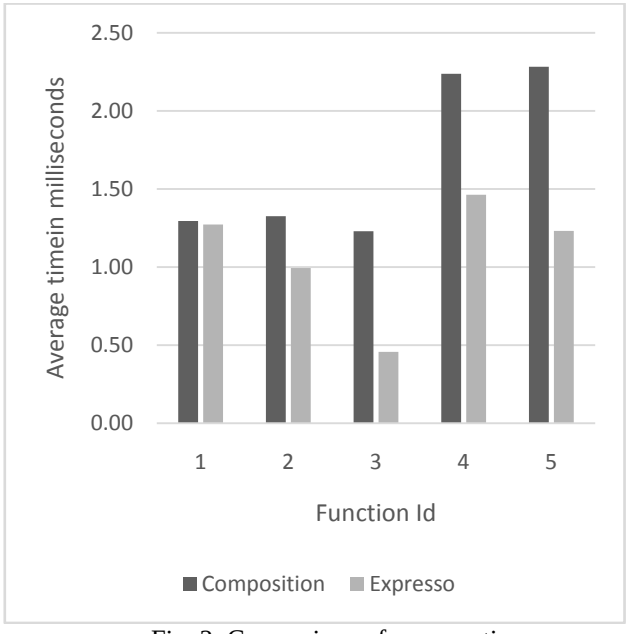

Fig. 2: Comparison of average time

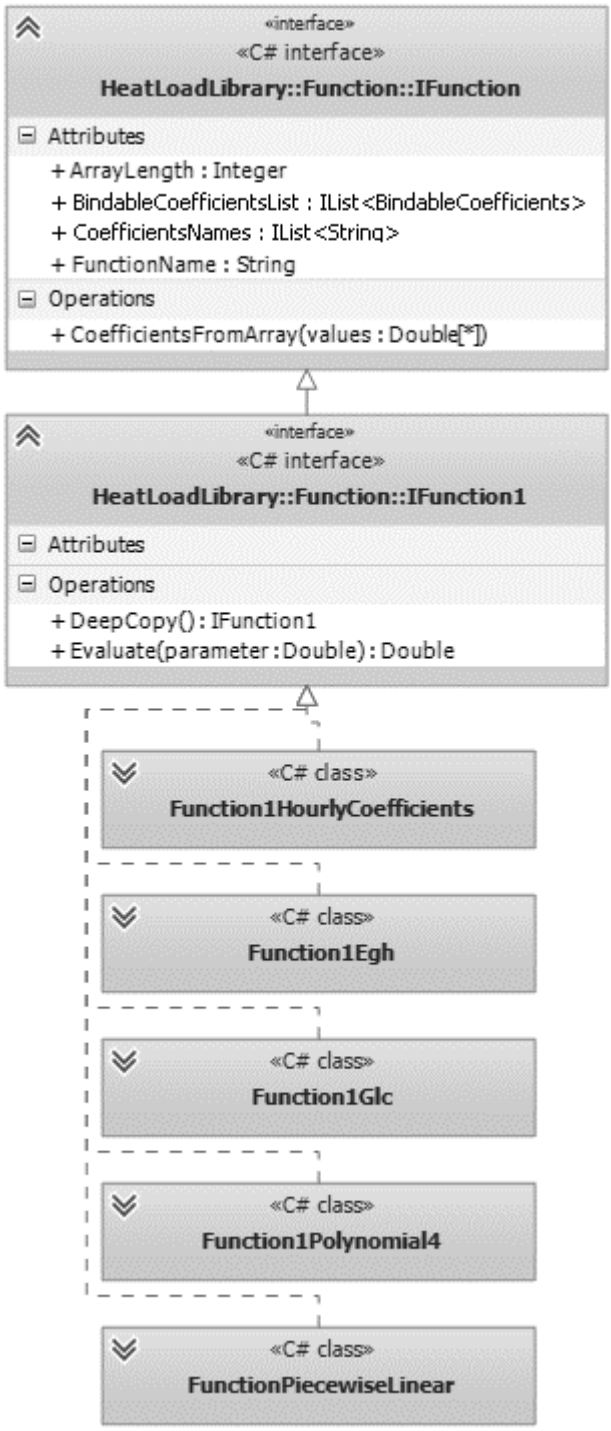

Fig. 3: Approximation function implementation

V. Conclusion

The main aim of this article is to compare the performance of parsed and dynamically compiled expressions with a static complied expression using a classic composite pattern.

The results were evaluated using the case example of heat load modelling. Users use this application as an excel add-in and the application is built using C# language and the .NET framework. Users can select combinations of different functions now or in the future. We compared the performance of two representations of functions used for heat load modelling. The former is a composition of a precompiled functions and the latter is a standard delegated using expression interpreter. The results show that the expression interpreter can performs better than the composition of precompiled functions.

References

[1] Erik Král. Additive and Multiplicative Heat Load Models Comparison. In Recent Advances in Circuits, Systems and Automatic Control. Budapešť: WSEAS press, 2013, s. 366-370. ISSN 1790-5117. ISBN 978-960-474-349-0.

[2] Petr Čápek, Erik Král. A Comparison of a Dynamic Compilation and Mathematic Parser Libraries in .NET for Expression Evaluation. Proceedings of Applied Mathematics, Simulation, Modelling (ASM '14). Italy, WSEAS press, Florence 2004.

[3] K. F. Chan and D. Y. Yeung, "Mathematical expression recognition: a survey," IJDAR, vol. 3, no. 1, pp. 3-15, 2000.

[4] PÁLSSON, Halldór, Helge LARSEN, Benny BØHM, Hans RAVN a JiJun ZHOU. Equivalent models of district heating systems for on-line minimization of operational costs of the complete district heating system. Lyngby: Department of Energy Engineering, Technical University of Denmark, 1999. ISBN 87-747-5221-9

[5] VAŘACHA, Pavel, JAŠEK, Roman. ANN Synthesis for an Agglomeration Heating Power Consumption Approximation. In Recent Researches in Automatic Control. Montreux: WSEAS Press, 2011, s. 239-244. ISBN 978-1-61804-004-6.

[6] HELLER, Alfred. Demand Modelling for Central Heating System. Lyngby, Denmark: Department of Buildings and Energy, Technical University of Denmark, 2000. ISBN 87-7877-042-4.

[7] WERNER, S.E. The Heat Load in District Heating System. Sweden: Chalmers University of Technology, 1984.

[8] Chramcov B.,Heat Demand Forecasting for Concrete District HeatingSystem. International Journal of Mathematical Models and Methods in Applied Sciences, WSEAS press, 2010, Vol. 4, No. 4, pp. 231-239.

[9] Dolinay V., Vašek L.,Simulation of Municipal Heating Network Based on Days with Similar Temperature, International Journal of Mathematics and Computers in Simulations, WSEAS press, 2011, Vol. 5, No. 5, pp. 470-477, ISSN 1998-0159.

[10] DOTZAUER, Erik. Simple model for prediction of loads in district-heating systems. Applied Energy. 2002, 73, s. 277-284

[11] Saarinen L. Modelling and control of a district heating system, Uppsala University, 2008, pp. 67 s., ISSN 1650-8300.

[12] TJØRVE, Even; TJØRVE, Kathleen. A unified approach to the Richards-model family for use in growth analyses: why we need only two model forms. Journal of theoretical biology, 2010, 267.3: 417-425.

[13] Jianwei Li, Comparison of the capability of peak functions in describing real chromatographic peaks, Journal of Chromatography A, Volume 952, Issues 1-2, 5 April 2002, Pages 63-70, ISSN 0021-9673, DOI: 10.1016/S0021-9673(02)00090-0.

[14] JKennedy J., Eberhart R., Particle Swarm Optimization IEEE International Conference on Neural Networks, Pert, WA, Australia, Nov. 1995, pp.1942-1948.

[15] Bratton D., Kennedy J., Defining a Standard for Particle Swarm Optimization, IEEE Swarm Intelligence Symposium, April 2007, pp.120-127.

[16] Zielinski K., Laur R., Stopping criteria for a constrained single-objective particle swarm optimization algorithm, Informatica, Vol. 31, No. 1, pp. 51-59, 2007.

[17] GitHub, DynamicExpresso [online] 2004, https://github.com/davideicardi/DynamicExpresso (Accessed: 20 October 2014).

Erik Král, Ph.D. is a senior lecturer at The Department of Computer and Communication Systems at The Tomas Bata University in Zlín. Between 2003 and 2006 he worked as a software developer (MS Navision DB, CRM system, .NET, c#). Between 2006 and 2011 he worked as a researcher on the National Research Program II, The intelligent system controlling an energetic framework of an urban agglomeration (successfully finished in 2011).

Petr Čápek is currently Ph.D. student at The Department of Informatics and Artifical Intelligence at Tomas Bata University in Zlín. His master's thesis was focused on the State-of-the-Art Methods for Designing Native Multiplatform Mobile Applications. Hi is interested in modern software architectures and evolution algorithms.