ANNALS

OF THE

UNIVERSITY OF CRAIOVA

Series: AUTOMATION, COMPUTERS, ELECTRONICS and MECHATRONICS

Vol. 12 (39), No. 1, 2015

ISSN 1841-0626

EDITURA UNIVERSITARIA

ANNALS OF THE UNIVERSITY OF CRAIOVA
Series: AUTOMATION, COMPUTERS, ELECTRONICS AND MECHATRONICS
Vol. 12 (39), No. 1, 2015 **ISSN 1841-0626**

***Note*:** The **"Automation, Computers, Electronics and Mechatronics Series"** emerged from **"Electrical Engineering Series" (ISSN 1223-530X)** in 2004.

Address for correspondence:
Emil PETRE
University of Craiova, Faculty of Automation, Computers and Electronics
Al.I. Cuza Street, No. 13, RO-200585, Craiova, Romania
Phone: +40-251-438198, Fax: +40-251-438198
Email: epetre@automation.ucv.ro

We exchange publications with similar institutions from country and from abroad

CONTENTS

ANNALS OF THE UNIVERSITY OF CRAIOVA
Series: Automation, Computers, Electronics and Mechatronics, Vol. 12 (39), No. 1, 2015

Method for Fault Detection

Eugen Iancu*, Sergiu Ivanov, Eugen Bobaşu*, Emil Petre***

**Department of Automation and Electronic, University of Craiova, 107 Decebal Street, RO-200440 Craiova, Romania (e-mail: Eugen.Iancu@automation.ucv.ro, http://www.ace.ucv.ro)*
***Department of Electromechanics, Environment and Industrial Informatics, University of Craiova, 107 Decebal Street, RO-200440 Craiova*

Abstract: This study shows a method for analytical fault detection that can be applied to monitor an electric motor brushless DC. The fault detection and the isolation (FDI) problem is an inherently complex one and for this reason the immediate goals is to preserve the stability of the process and, if is possible, to control the process in a slightly degraded manner. The authors propose a practical method to detect the presence of failures of sensors using a prediction solution. It was used for this purpose a mathematical model of BLDC motor and single exponential smoothing. Also is proposed a structure to detect the presence of failures.

Keywords: Brushless DC motor, fault detection and isolation, analytical redundancy, single exponential smoothing.

1. INTRODUCTION

Changes (faults) can make the system unsafe and less reliable. Productivity of the automatic system can degrade because changes can impose performance limitations on the system and may also require frequent system shut downs for its maintenance. In order to avoid production deteriorations or damage to machines and humans, variations have to be detected as quickly as possible and decisions that stop the propagation of their effects have to be made (Nguang et al., 2005).

The necessity to obtain a diagnostic with good performances, without installing a lot of redundant and dedicated expensive equipment, forces the diagnostic tools to develop the techniques available to processing all the information that are "hidden" in the technological process. In fact which the reality of industrial systems can offer to the engineer charged to implement the monitoring functions, is usually very inadequate: poor models available, lack of redundancy, insufficient number of measures, noise on the acquired data, unmodeled disturbances, etc.

The problem of reliable fault diagnosis in dynamic processes has received great attention and a wide variety of robust approaches has been proposed and developed. Analytical redundancy–based methods have been developed to diagnose faults in linear, time invariant, dynamic systems and a wide variety of model–based approaches has been proposed (Chen and Patton, 1999).

All failure detection methods exploit redundant data, which are obtained either directly, when two or more sensors are available for measurement of a process variable, or analytically, when a process variable is estimated using the mathematical process model. These redundancy relationships may then be exploited to generate residual signals. Under normal operating conditions these residuals are "*small*" in an appropriate sense and yet display distinct patterns when failures occur.

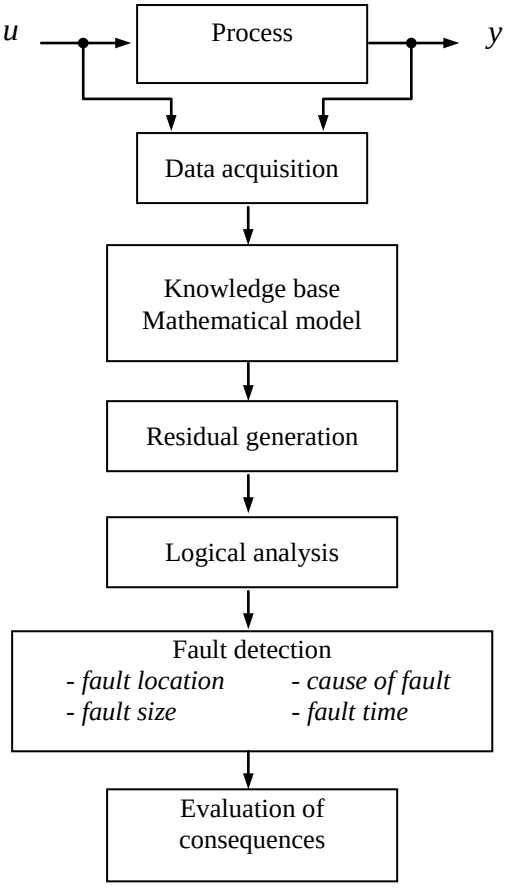

Fig. 1. The structure of the analytical diagnostic (Iancu and Vinatoru, 2005).

The failure diagnosis process consists in three stages (Fig. 1):

- Modeling of process
- Residual generation
- Residual analysis.

The residuals must be carefully examined to determine the presence of failures (detection) and which system components have failed (isolation). In practice it is often difficult to fulfil the demands of the method for the complex diagnosed plant. Robust methods of diagnosis are therefore required, in the face of existing measurement uncertainty, disturbances and incomplete knowledge (Patton, 1994). In such cases an integrated approach using quantitative and qualitative models in diagnostic expert systems gives a good solution.

2. MATHEMATICAL MODEL OF BRUSHLESS DIRECT CURRENT MOTOR

Brushless DC motor (BLDC) is a electric motor, type synchronous, usually three-phase, having rotor with permanent magnets and stator built with concentrated or uniformly distributed windings. Specific electronic circuit is accomplished by switching semiconductor elements of the three-phase power inverter that is synchronized with the rotor position. Rotor position must be known at specific angles to align with the applied electric voltage. Usually, in order to obtain information on the rotor possition are using three Hall effect sensors, encapsulated in stator.

The mathematical model of the drive can be made using the principle ilustrated in the equivalent scheme represented in Fig.2.

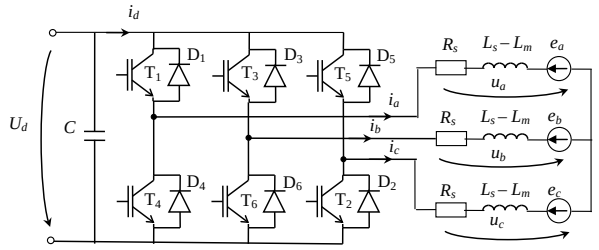

Fig. 2. Schematic diagram of the drive with brushless DC motor.

The equations of the phase voltages, are (Kennedy and Eberhart, 1995; Fedák et al., 2012):

$$u_a = R_s i_a + L_s \frac{di_a}{dt} + L_m \frac{di_b}{dt} + L_m \frac{di_c}{dt} + e_a, \quad (1)$$

$$u_b = R_s i_b + L_m \frac{di_a}{dt} + L_s \frac{di_b}{dt} + L_m \frac{di_c}{dt} + e_b, \quad (2)$$

$$u_c = R_s i_c + L_m \frac{di_a}{dt} + L_m \frac{di_b}{dt} + L_s \frac{di_c}{dt} + e_c, \quad (3)$$

where:

- u_a, u_b, u_c – instantaneous values of the phase voltages;
- e_a, e_b, e_c – the instantaneous electric phase voltages;
- i_a, i_b, i_c – the instantaneous phase currents;
- R_s, L_s – phase stator resistance and inductance, assumed the same on all phases;
- L_m – the mutual inductance.

Given the star connection of the stator windings, provided

$$i_a + i_b + i_c = 0 \quad (4)$$

allows writing equations (1) - (3) as:

$$u_a = R_s i_a + \left(L_s - L_m\right)\frac{di_a}{dt} + e_a; \quad (5)$$

$$u_b = R_s i_b + \left(L_s - L_m\right)\frac{di_b}{dt} + e_b; \quad (6)$$

$$u_c = R_s i_c + \left(L_s - L_m\right)\frac{di_c}{dt} + e_c. \quad (7)$$

Electric phase voltages depend by the position of the rotor to form a balanced three-phase system, the waveform imposed by the trapezoidal rule (Fig. 3). Thus, they can be expressed as (Fedák et al., 2012; Prasad et al., 2012):

$$e_a = \omega \cdot K_e \cdot f\left(\theta_e\right), \quad (8)$$

$$e_b = \omega \cdot K_e \cdot f\left(\theta_e - 2\pi/3\right), \quad (9)$$

$$e_c = \omega \cdot K_e \cdot f\left(\theta_e - 4\pi/3\right), \quad (10)$$

where θ_e is the electric angle of the motor, ω is the angular speed of the rotor, K_e is constant for electromotive voltages (V/(rad / sec)) and the reference function $f(\theta_e)$ for electric tension is alternatively trapezoidal with amplitude 1 (Fig. 3).

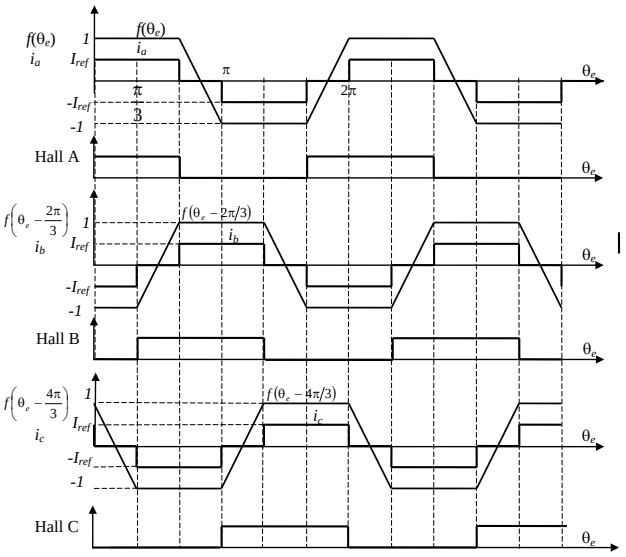

Fig. 3. Electromotive voltages, phase currents and position transducer outputs Hall

Phase alternating currents are rectangular, with alternations during the $2\pi/3$ centered on the electric voltages (Fig. 3).

Taking the origin of the phase as shown in Fig. 3, the function $f(\theta_e)$ defined on intervals has the next expression:

$$f(\theta_e)=\begin{cases} 1 & \text{for } \theta_e \in \left[0,\frac{2\pi}{3}\right] \\ 1-\frac{6}{\pi}\left(\theta_e-\frac{2\pi}{3}\right) & \text{for } \theta_e \in \left[\frac{2\pi}{3},\pi\right] \\ -1 & \text{for } \theta_e \in \left[\pi,\frac{5\pi}{3}\right] \\ -1+\frac{6}{\pi}\left(\theta_e-\frac{5\pi}{3}\right) & \text{for } \theta_e \in \left[\frac{5\pi}{3},2\pi\right] \end{cases} \quad (11)$$

To the voltage equations are added the equation of motion

$$m = J\cdot\frac{d\omega}{dt}+B\cdot\omega+m_s \quad (12)$$

where m is the electromagnetic torque developed by the motor, m_s is the static torque, J is the total moment of inertia (motor plus load), and B is the friction coefficient.

Also, the electromagnetic torque developed by the motor has the expression (Kennedy and Eberhart, 1995):

$$m=\frac{e_a i_a+e_b i_b+e_c i_c}{\omega}= K_e\left[i_a f(\theta_e-\pi/3)+i_b f(\theta_e-2\pi/3)+i_c f(\theta_e-4\pi/3)\right] \quad (13)$$

Torque is mainly influenced by the waveforms of the electromotive voltages induced in the stator of electric motors due to rotor motion. Ideally, electric tensions have trapezoidal waveforms and stator currents are rectangular (Fig. 3), resulting a constant torque. In practice, there are pulsations of torque due to design imperfections that lead to removal from electric tensions perfect form trapezoidal or due to PWM control method, switching and hysteresis. Taking into account of connection between electrical angle θ_e, mechanically angle θ_m, and the number of pole pair's p, we have the next relation:

$$\theta_e = p\cdot\theta_m, \quad (14)$$

and that

$$\omega=\frac{d\theta_m}{dt}, \quad (15)$$

Operating equations (5) - (7), (12) and (15) can be written as the state equations, respectively:

$$\frac{di_a}{dt}=-\frac{R_s}{L_s-L_m}i_a-\frac{K_e f(\theta_e)}{L_s-L_m}\omega+\frac{1}{L_s-L_m}u_a \quad (16)$$

$$\frac{di_b}{dt}=-\frac{R_s}{L_s-L_m}i_b-\frac{K_e f(\theta_e-2\pi/3)}{L_s-L_m}\omega+\frac{1}{L_s-L_m}u_b \quad (17)$$

$$\frac{di_c}{dt}=-\frac{R_s}{L_s-L_m}i_c-\frac{K_e f(\theta_e-4\pi/3)}{L_s-L_m}\omega+\frac{1}{L_s-L_m}u_c \quad (18)$$

$$\frac{d\omega}{dt}=\frac{K_e f(\theta_e)}{J}i_a+\frac{K_e f(\theta_e-2\pi/3)}{J}i_b+ +\frac{K_e f(\theta_e-4\pi/3)}{J}i_c-\frac{B}{J}\omega-\frac{1}{J}m_s \quad (19)$$

$$\frac{d\theta_e}{dt}=p\omega\cdot \quad (20)$$

In the form of a matrix, the equation of state is:

$$\dot{\xi}=\mathbf{A}\cdot\xi+\mathbf{B}\cdot\mathbf{u}; \quad (21)$$

the vectors of inputs (u) and state variables (ξ) are:

$$\mathrm{u}=\begin{bmatrix}u_a & u_b & u_c & m_s\end{bmatrix} \quad (22)$$

$$\xi=\begin{bmatrix}i_a & i_b & i_c & \omega & \theta_e\end{bmatrix} \quad (23)$$

and the matrices **A** and **B** have the form:

$$\mathbf{A}=\begin{bmatrix} -\frac{R_s}{L_s-L_m} & 0 & 0 & -\frac{K_e f(\theta_e)}{L_s-L_m} & 0 \\ 0 & -\frac{R_s}{L_s-L_m} & 0 & -\frac{K_e f(\theta_e-2\pi/3)}{L_s-L_m} & 0 \\ 0 & 0 & -\frac{R_s}{L_s-L_m} & -\frac{K_e f(\theta_e-4\pi/3)}{L_s-L_m} & 0 \\ \frac{K_e f(\theta_e)}{J} & \frac{K_e f(\theta_e-2\pi/3)}{J} & \frac{K_e f(\theta_e-4\pi/3)}{J} & -\frac{B}{J} & 0 \\ 0 & 0 & 0 & p & 0 \end{bmatrix} \quad (24)$$

$$\mathbf{B}=\begin{bmatrix} \frac{1}{L_s-L_m} & 0 & 0 & 0 \\ 0 & \frac{1}{L_s-L_m} & 0 & 0 \\ 0 & 0 & \frac{1}{L_s-L_m} & 0 \\ 0 & 0 & 0 & -\frac{1}{J} \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (25)$$

3. THE GENERATION OF RESIDUE DURING THE MODEL-BASED DIAGNOSIS

When the system has the actuators affected by faults, this situation can be described by the relation:

$$\begin{cases}\dot{\boldsymbol{x}}(\boldsymbol{t})=\mathbf{A}\boldsymbol{x}(\boldsymbol{t})+\mathbf{B}\boldsymbol{u}(\boldsymbol{t})+\mathbf{B}\boldsymbol{u}_{\boldsymbol{d}}(\boldsymbol{t}) \\ \boldsymbol{y}(\boldsymbol{t})=\mathbf{C}\boldsymbol{x}(\boldsymbol{t})+\mathbf{D}\boldsymbol{u}(\boldsymbol{t})+\mathbf{D}\boldsymbol{u}_{\boldsymbol{d}}(\boldsymbol{t})\end{cases} \quad (26)$$

where $x(t)$ is the system's state, $y(t)$ is the system's output, $u(t)$ is the system's command, **A**, **B, C** and **D** are constant matrices of appropriate dimensions and the vector $u_d(t)$ represent the fault vectors for the actuators. The transfer function type input-output representation for the system is the following:

$$y(s) = H(s)u(s) + H(s)u_d(s) \quad (27)$$

where

$$H(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} \quad (28)$$

The residue generator is a linear processor whose inputs consist in the input and output of the monitored system. This structure can be expressed mathematically so (Fig. 4):

$$r(s) = \left[P(s)Q(s)\right] \cdot \begin{bmatrix} u(s) \\ y(s) \end{bmatrix} = P(s)u(s) + Q(s)y(s) \quad (29)$$

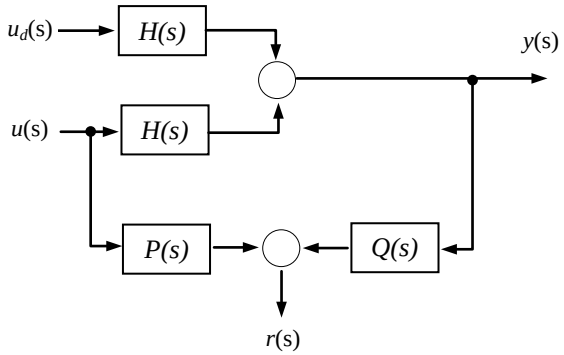

Fig. 4. The generation of the residue.

The matrixes $P(s)$ and $Q(s)$ are transfer matrixes built using linear, stabile systems. According to the definition, the residue is designed to become 0 in the case of fault absence and different from 0, in the presence of faults.

$$r(t) = 0 \text{ if and only if } u_d(t) = 0 \quad (30)$$

For the residue generator $r(s)$ to be a fault indicator, the transfer matrixes $P(s)$ and $Q(s)$ must satisfy the relation:

$$P(s) + Q(s)H(s) = 0 \quad (31)$$

When the system has the sensors affected by faults, this situation can be expressed using next structure (Fig. 5).

Let to consider a dynamical process of the form given by the linearised equations:

$$\dot{x}(t) = \mathbf{A}x(t) + \mathbf{B}u_c(t) \quad (32)$$

where $x \in \Re^n$ is the state vector, $u \in \Re^m$ is the known input vector, and **A**, **B** are constant matrices of appropriate dimensions. Similarly, the mathematical model of the process is:

$$\dot{x}_m(t) = \mathbf{A}x_m(t) + \mathbf{B}u_c(t) \quad (33)$$

The residual vector $r(t)$ is generated by the equation:

$$r(t) = x(t) - x_m(t) \quad (34)$$

$$r(t) = 0 \text{ if and only if } u_f(t) = 0 \quad (35)$$

Ideally, in absence of a fault, the residual should be zero, while when a fault is present it should be different from zero. So, a fault detection test will consist in check if the residual is zero or not.

4. EXPONENTIAL SMOOTHING METHOD

Single exponential smoothing is used for smoothing discrete time series. The efficiency of this algorithm can be attributed to its simplicity and to the capacity to adjust its responsiveness to changes in the process and its reasonable accuracy.

Let be an observed time series $X = \{x_1 \quad x_2 \quad \ldots \quad x_n\}$. Formally, the simple exponential smoothing equation takes the form (Ostertagová, 2011):

$$\tilde{x}_{i+1} = \alpha x_i + (1-\alpha)\tilde{x}_i \quad (36)$$

where x_i is the actual, known series value at moment time i, $\tilde{x}_i$ is the forecast value of the variable X at time i, $\tilde{x}_{i+1}$ is the forecast value at time i+1 and α is the smoothing constant.

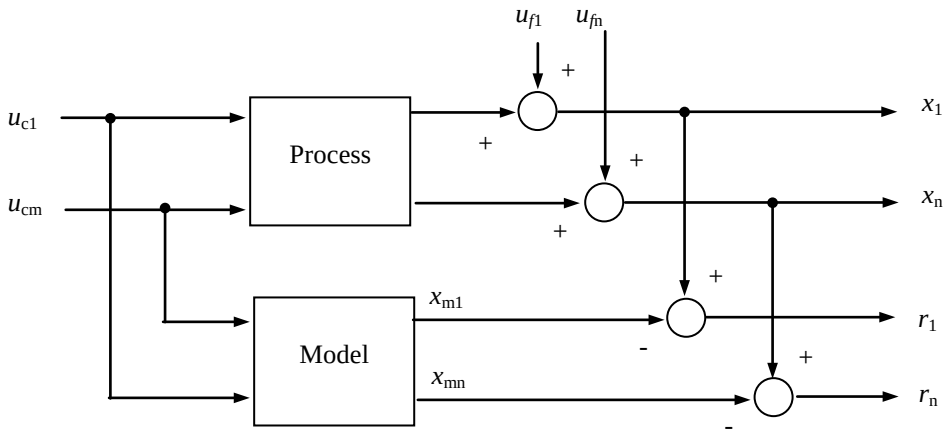

Fig. 5. Method for generates the residual vector for the sensor diagnosis.

Smoothing constant α is a selected number between zero and one, $0 < \alpha < 1$ (Brown and Meyer, 1961). When $\alpha = 1$, the original and smoothed version of the series are identical. At the other extreme, when $\alpha = 0$, the series is smoothed flat (Ostertagová, 2011). In the literature it is demonstrate the next relation (Brown and Meyer, 1961):

$$\tilde{x}_{i+1} = \alpha x_i + \alpha(1-\alpha)x_{i-1} + \alpha(1-\alpha)^2 x_{i-2} + ... \\ ... + \alpha(1-\alpha)^{i-1} x_1 = \alpha \sum_{k=0}^{i-1} (1-\alpha)^k x_{i-k} \quad (37)$$

In scientific papers are presented also *double exponential smoothing* and *triple exponential smoothing*.

From (36) we obtain:

$$\tilde{x}_{i+1} = \tilde{x}_i + \alpha(x_i - \tilde{x}_i) = \tilde{x}_i + \alpha\varepsilon_i \quad (38)$$

where ε_i represent the forecast error at time *i*. Using this error it is possible to define the following parameters (Ostertagová, 2011):

- Mean square error - *MSE*

$$MSE = \frac{1}{n}\sum_{i=1}^{n} e_i^2 \quad (39)$$

- Root mean square error - *RMSE*

$$RMSE = \sqrt{MSE} \quad (40)$$

The objective is to find an appropriate smoothing constant so that *MSE* and *RMSE* to be minimum.

5. ANALYTICAL METHOD FOR DETECTION OF FAILED SENSOR

Let to consider a dynamical process of the form given by the linearised equations (32). The method proposed by the authors is ilustred in Fig. 6. The principle is:

Step 1: The residual vector is generated.

Step 2: Using the exponential smoothing method it is calculate an anticipative value of the residual vector.

Step 3: The anticipative residual vector is used to generate an alarm (if it is different from zero) and allows the location of the faulty sensor.

6. CONCLUSIONS

The proposed method is an analytical solution, which represent a new approach for the computer-assisted diagnosis. The presented algorithm is very useful when the parameters of the process are affected by small modifications, progressive degradation, that influences the functions of the system in a slightly degraded manner.

ACKNOWLEDGMENT

This paper was supported by Project no. P09004/1137/31.03.2014, cod SMIS 50140, entitled: "Industrial research and experimental development vehicles powered by brushless electric motor supplied by lithium-ion accumulators for people transport - ELECTRIC GENTLE".

REFERENCES

Brown, R.G. and Meyer, R.F. (1961). The fundamental theory of exponential smoothing, *Operations Research*, vol. 9, pp. 673-685.

Chen, J. and Patton R.J. (1999). *Robust Model–Based Fault Diagnosis for Dynamic Systems*, Kluwer Academic Publishers.

Fedák, V., Balogh, T., and Záskalický, P. (2012). *Dynamic Simulation of Electrical Machines and Drive Systems Using MATLAB GUI - A Fundamental Tool for Scientific Computing and Engineering Applications* – Vol. 1, Vasilios Katsikis, InTech. Ed.

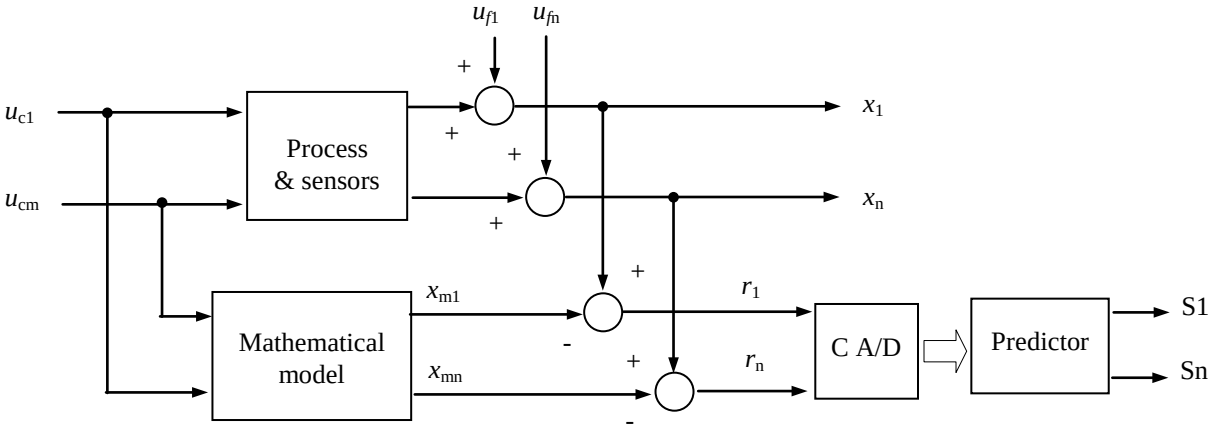

Fig. 6. Method for generates the residual vector for the sensor diagnosis.

Prasad, G., Sree Ramya, N., Prasad, P.V.N., and Tulasi Ram Das, G. (2012). Modelling and Simulation Analysis of the Brushless DC Motor by using MATLAB, *Int. J. of Innovative Technology and Exploring Engineering*, Vol. 1, Issue 5, Oct. 2012.

Iancu E. and Vinatoru M. (2005). A fault detection and isolation system using neural networks, *Proceedings of the International Conference on Control Systems and Computer Science CSCS 15*, Bucuresti, vol. 1, pp. 422-427.

Kennedy, J. and Eberhart, R.C. (1995). Particle swarm optimization. *Proc. of IEEE Int. Conference on Neural Networks*, Piscataway, NJ. pp. 1942-1948.

Nguang S.K., Shi P., Ding S. (2005). Fault detection filter for uncertain fuzzy systems: an LMI approach, *IFAC Congress*, Praha, CD proceedings.

Ostertagová, E. and Ostertag O. (2011). The simple exponential smoothing model, *Proceedings of the 4th International Conference on Modelling of Mechanical and Mechatronic Systems*, Technical University of Košice, Slovak Republic, , pp. 380-384.

Patton R.J. and Chen J. (1994). A review of parity space approaches to fault diagnosis applicable to aerospace systems, *Journal of Guidance, Control and Dynamics*, vol. 17, no. 2, pp. 278-285.

ANNALS OF THE UNIVERSITY OF CRAIOVA
Series: Automation, Computers, Electronics and Mechatronics, Vol. 12 (39), No. 1, 2015

Fault Detection in Educational Kit Festo

Camelia Maican*, Gabriela Cănureci**

** Automation and Electronics Department, University of Craiova, Romania (e-mail: camelia@automation.ucv.ro).*
*** Research and Development, Engineering and Manufacturing for Automation Equipment and Systems SC IPA SA CIFATT Craiova, Romania (e-mail: gabriela.canureci@ipacv.ro).*

Abstract: In this paper is study the faults detection and localization, in educational kit Festo, using residual methods. The level control system and the faults detection structure were developed under Matlab Simulink. This faults detection structure allows us to detect two faults that can occur in the plant, separately and simultaneously. The proposed method was theoretically developed and experimentally verified on the plant model.

Keywords: control; fault detection and localization; level; residue.

1. INTRODUCTION

The plant Festo contains individual modules which can be combined in different ways and allows the level, temperature and flow control. Based on equivalence relations of flowing phenomena, one can determine the equivalence of the level controlling system within Festo plant with the level controlling system of the drums in the steam boilers within the thermal power groups (Canureci et al. 2012).

Within both plants, there can be malfunctions like:

- blocking the adjustment flap of the boiler feed water flow, or a changing transfer factor, which can be simulated on the Festo plant by modifying the flowing section of the tap on the circulation pump discharge.

- pipe breaking or stopping of a power pump on the water flow of the boiler, which lowers the boiler feed water flow according to the value generated by the controller, fact that can be simulated on the plant by activating the tap R_3, which partially controls the flow Fp directly to the second tank.

This way, the structures of detection and identification of the malfunctions that may appear on the given plant can be applied analogically to the steam boiler level control system (Iancu et al. 2003 and Vinatoru 2001).

2. THE MATHEMATICAL MODELS

The hydraulic system diagram of the educational kit FESTO is shown in Fig.1 (Canureci et al. 2012). The plant consists of two parallelepipedic transparent plastic water tanks assembled on an aluminium platform with supporting holders. The tanks are placed one in upper position the other in lower position. A water pump (P) ran by a driving motor (DM) ensures water flow from the lower tank to the upper tank through a system of pipelines, bends and connecting pipes.

At the exit of the pump (P), on the pipe there is a vertically assembled hydraulic diode of connection which prevents water leaks from the upper tank to the lower tank when the pump discharge pressure gets below the hydrostatic pressure which corresponds to the liquid level of the upper tank. A pipe system combined with two taps R_1 and R_2 allow conducting the water discharged by the pump either towards the upper tank or to the lower tank. Water flow from the upper to the lower tank is done naturally through a pipe system on which there is a tap R_3 which allows changes in the flowing section or obstructions of the pipe (Vinatoru et al. 2008).

On the upper tank there is mounted a level transducer with ultrasounds which determines an electric signal at the exit 4-20 mA DC for a fluctuation of the liquid level within the field 0-500 mm (Vinatoru et al. 2007). The liquid level control is achieved by changing the water pump flow ($F_3=F_p$), using the speed command of the driving motor (DM).

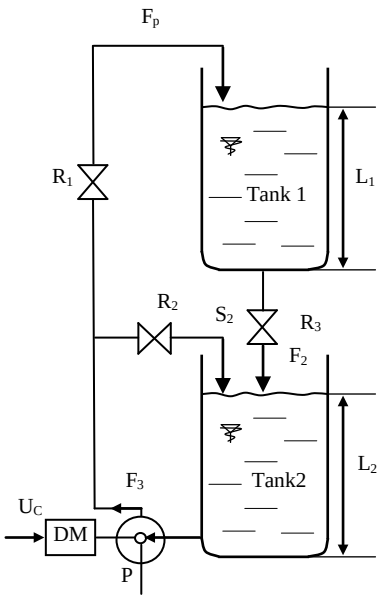

Fig. 1 The hydraulic system diagram

This is achieved by varying the supply voltage of the pump using the pump regulator, which is driven with a voltage signal in the range of 2-10 V DC.

The state variables are:

$x_1 = L_1$ the level of the tank1

$x_2 = L_2$ the level of the tank2

The input variables are:

$F_3 = F_p = U = k_p U_c$ the command to adjust the level in the tank 1

$F_2 = C_1 S_2 \sqrt{\rho g L_1} = C S_2 \sqrt{L_1}$ the evacuation flow from the tank 2.

The mathematical model of the system is determined using the mass balance equations (Vinatoru et al. 2008):

$$\frac{dL_1}{dt} = \frac{F_p(U_c) - F_2(S_2, L_1)}{A_1} = \frac{k_p U_C - C\ S_2\sqrt{L_1}}{A_1} \quad (1)$$

$$\frac{dL_2}{dt} = \frac{F_2(S_2, L_1) - F_3(U_C)}{A_2} = \frac{C\ S_2\sqrt{L_1} - k_p U_C}{A_2} \quad (2)$$

where C is a coefficient depending on the viscosity of the fluid loss and sectional shape of the flowing, S_2 sectional area of the valve transitions between the two tanks, A_1 and A_2 cross-sectional area of the Tank1, respectively Tank2.

In canonical form the mathematical model is:

$$\frac{dL_1}{dt} = -\frac{C}{A_1} S_2\sqrt{L_1} + \frac{K_P}{A_1} U_C \quad (3)$$

$$\frac{dL_2}{dt} = \frac{C}{A_2} S_2\sqrt{L_1} - \frac{K_P}{A_2} U_2 \quad (4)$$

In steady state:

$$k_p U_{P0} = C\ S_{20}\sqrt{L_{10}} = F_{20} \quad (5)$$

By linearizing of the equations (3) and (4) around the steady state values, resulting linear form of the mathematical model (6) and (7), which contain the section variation ΔS_2 which can occur only under fault conditions. Under normal conditions $\Delta S_2 = 0$.

$$\Delta\dot{x}_1 = -\frac{CS_{20}}{2A_1\sqrt{L_{10}}}\Delta x_1 + \frac{k_p}{A_1}\Delta U_C - \frac{C\sqrt{L_{10}}}{A_1}\Delta S_2 \quad (6)$$

$$\Delta\dot{x}_2 = \frac{CS_{20}}{2A_1\sqrt{L_{10}}}\Delta x_1 - \frac{k_p}{A_1}\Delta U_C + \frac{C\sqrt{L_{10}}}{A_1}\Delta S_2 \quad (7)$$

where: $\Delta x_1 = \Delta L_1 = L_1 - L_{10}$

$\Delta x_2 = \Delta L_2 = L_2 - L_{20}$,

$x_{10} = L_{10}$ and $x_{20} = L_{20}$

Applying Laplace transform in zero initial conditions result:

$$\Delta x_1(s) = \frac{1}{Ts+1}\left[\frac{2x_{10}k_p}{F_{20}}\Delta U_C(s) - \frac{2x_{10}}{S_{20}}\Delta S_2(s)\right] \quad (8)$$

$$\Delta x_2(s) = \frac{A_1/A_2}{Ts+1}\left[-\frac{2x_{10}k_p}{F_{20}}\Delta U_C(s) + \frac{2x_{10}}{S_{20}}\Delta S_2(s)\right] \quad (9)$$

where: $$T = \frac{2A_1 x_{10}}{F_{20}}$$

and $A_1 = A_2 = 1{,}75 \cdot 1{,}9 = 3{,}325\text{dm}^2$

$L_{10} = L_{20} = 2\text{dm}$

$F_{10} = F_{20} = 4{,}7\text{dm}^3$

3. THE FAULT DETECTION AND LOCALIZATION

We will analyze the following faults which may occur in the FESTO system:

- The pump is not running completely the command received or an additional pressure loss occurs on the above valve, which is mounted after the pump, and this makes the pump flow to be influenced by fault, therefore be no longer proportional to the control voltage:

$$\Delta F_p = k_p\left(\Delta U_{Cn} + \Delta U_d\right) \quad (10)$$

where ΔU_{Cn} is pump control signal and ΔU_d the equivalent in the control signal of the actuator's fault.

- Modification of the flow section S_2 due either to variation of the flow regime, by plugging the crossing section or additional breakings of pipeline, which is equivalent to an increase of S_2; this makes as the exhaust flow from the Tank1 in Tank 2 to be of the form:

$$\Delta F_2 = C\sqrt{L_{10}}\left(\Delta S_{2n} + \Delta S_{2d}\right) \quad (11)$$

Where ΔS_{2n} is the value of the section under normal functioning and ΔS_{2d} is equivalent value of the section, caused by the fault.

In these circumstances, by introducing the faults specified in equations (6) and (7), replacing

$$\Delta U_C = \Delta U_{Cn} + \Delta U_d$$

$$\Delta S_2 = \Delta S_{2n} + \Delta S_{2d}$$

we obtain:

$$\Delta\dot{x}_1 = -\frac{CS_{20}}{2A_1\sqrt{L_{10}}}\Delta x_1 + \frac{k_p}{A_1}\left(\Delta U_C + \Delta U_d\right) - \\ -\frac{C\sqrt{L_{10}}}{A_1}\left(\Delta S_{2n} + \Delta S_{2d}\right) \quad (12)$$

$$\Delta\dot{x}_2 = \frac{CS_{20}}{2A_1\sqrt{L_{10}}}\Delta x_1 - \frac{k_p}{A_1}(\Delta U_C + \Delta U_d) + \\ + \frac{C\sqrt{L_{10}}}{A_1}(\Delta S_{2n} + \Delta S_{2d}) \quad (13)$$

Applying Laplace in equations (12) and (13) in zero initial conditions and using the transformations and the notations from equations (8) and (9) it follows the operational form of the real process equations in which faults were included.

Following the same steps to equations (12) and (13) as the relations (6) and (7) it results:

$$x_{1r}(s) = \frac{1}{Ts+1}\left[\begin{array}{l}\frac{2x_{10}k_p}{F_{20}}\Delta U_{Cn}(s) + \frac{2x_{10}k_p}{F_{20}}\Delta U_d(s) - \\ -2x_{10}\frac{\Delta S_{2n}(s)}{S_{20}} - 2x_{10}\frac{\Delta S_{2d}(s)}{S_{20}}\end{array}\right] \quad (14)$$

$$x_{2r}(s) = \frac{A_1/A_2}{Ts+1}\left[\begin{array}{l}-\frac{2x_{10}k_p}{F_{20}}\Delta U_{Cn}(s) - \frac{2x_{1r}k_p}{F_{20}}\Delta U_d(s) + \\ +2x_{10}\frac{\Delta S_{2n}(s)}{S_{20}} + 2x_{10}\frac{\Delta S_{2d}(s)}{S_{20}}\end{array}\right] \quad (15)$$

where: $$\frac{CS_{20}}{2A_1\sqrt{L_{10}}} = \frac{1}{T}$$

From equations (8) and (9) it follows the process model, considering $\Delta S_{2d} = 0$:

$$x_{1m}(s) = \frac{1}{Ts+1}\left[\frac{2x_{10}k_p}{F_{20}}\Delta U_{Cn}(s) - 2x_{10}\frac{\Delta S_{2n}(s)}{S_{20}}\right] \quad (16)$$

$$x_{2m}(s) = \frac{1}{Ts+1}\left[-\frac{2x_{10}k_p}{F_{20}}\Delta U_{Cn}(s) + 2x_{10}\frac{\Delta S_{2n}(s)}{S_{20}}\right] \quad (17)$$

We define the residues $r_1(s)$ and $r_2(s)$ (Gertler et al. 2002 and Korbicz et al. 2004):

$$r_1(s) = x_{1r}(s) - x_{1m}(s)$$

$$r_2(s) = x_{2r}(s) - x_{2m}(s)$$

$$r_1(s) = \frac{1}{(Ts+1)}\frac{2x_{10}k_p}{F_{20}}U_d(s) - \frac{1}{(Ts+1)}2x_{10}\frac{\Delta S_{2d}(s)}{S_{20}} \quad (18)$$

$$r_2(s) = -\frac{1}{Ts+1}\frac{2x_{10}k_p}{F_{20}}\Delta U_d(s) + \frac{1}{Ts+1}2x_{10}\frac{\Delta S_{2d}(s)}{S_{20}(s)} \quad (19)$$

From the equations (18) and (19) it can be seen that the residues r_1 and r_2 are both of influenced by the faults ΔS_{2d} and ΔU_d (Table.1).

Table 1. Influence of faults to the residues

Residues/faults	ΔU_d	ΔS_{2d}
r_1	1	1
r_2	1	1

In this case, the fault matrix is:

$$G_D(s) = \begin{bmatrix} \frac{2x_{10}k_p}{F_{20}(Ts+1)} & -\frac{2x_{10}}{Ts+1} \\ -\frac{2x_{10}k_p}{F_{20}(Ts+1)} & \frac{2x_{10}}{Ts+1} \end{bmatrix} \quad (20)$$

where: $$\frac{2\cdot x_{10}\cdot k_p}{F_{20}} = \frac{2\cdot 2\cdot 1{,}4}{4{,}7} = 1{,}2 \text{ [dm/V]}$$

$$T = \frac{2A_1x_{10}}{F_{20}} = 2{,}8 \text{ [min]}$$

$$G_D(s) = \begin{bmatrix} \frac{1{,}2}{2{,}8s+1} & -\frac{4}{2{,}8s+1} \\ -\frac{1{,}2}{2{,}8s+1} & \frac{4}{2{,}8s+1} \end{bmatrix} \quad (21)$$

From the equation (21) it can observe that the fault matrix is non-invertible, because $\det G_D(s) = 0$.

Accordingly, we use only the residue r1 with which we can determine the fault ΔU_d or the fault ΔS_{2d}.

Because: $$G_D(s) = \frac{1{,}2}{2{,}8s+1}$$

Result: $$G_D^{-1}(s) = \frac{2{,}8s+1}{1{,}2}$$

Because $G_D^{-1}(s)$ cannot be physically realized, for implementation, is introduces a pole, with a small time constant, and in this case is chosen:

$$G_D^{-1*}(s) = \frac{2{,}8s+1}{1{,}2(s+1)} \quad (22)$$

To detect and locate the faults was achieved the simulation scheme shown in Fig.2, with the following blocks:

- In the upper part is simulated the Festo system, with the level control structure of the Tank 2 commanding of the pump flow, which feed the Tank 1. In the system are simulated the fault ΔU_d representing the variation of pump flow and the fault ΔS_{2d} representing an additional flow variation drain from Tank 1.
- In the lower part is the Festo model, functioning in normal condition, to which the same command provided by PI controller is applied.

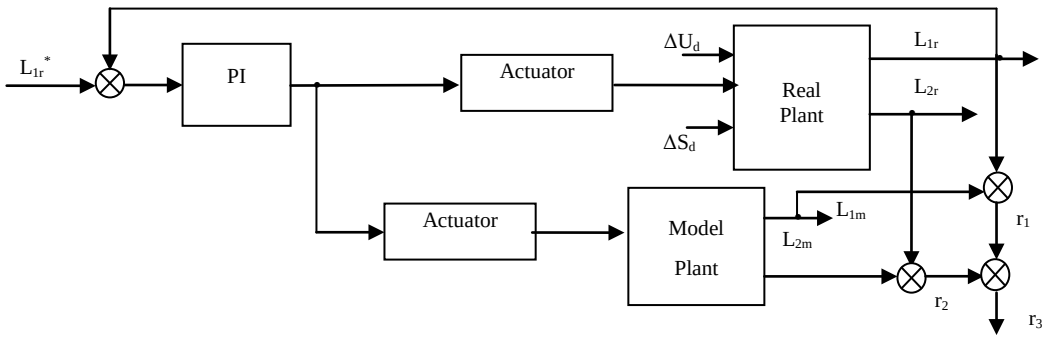

Fig. 2 The fault detection structure

Also in the lower part were generated the residues:

$$r_1=L_{1r}-L_{1m}$$
$$r_2=L_{2r}-L_{2m} \quad (23)$$
$$r_3=r_1+r_2$$

Simulation results are presented in the next figures.

In Fig 3and Fig. 4 are represented the level for the real plant, respective the residues in normal condition.

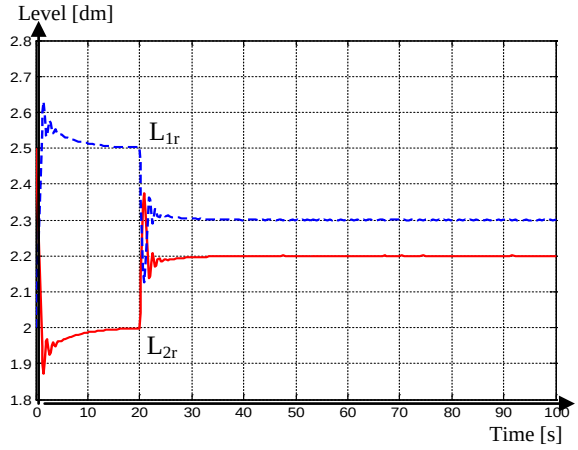

Fig. 3 The level L_{1r} and L_{2r} in normal condition to the change of the prescribed size

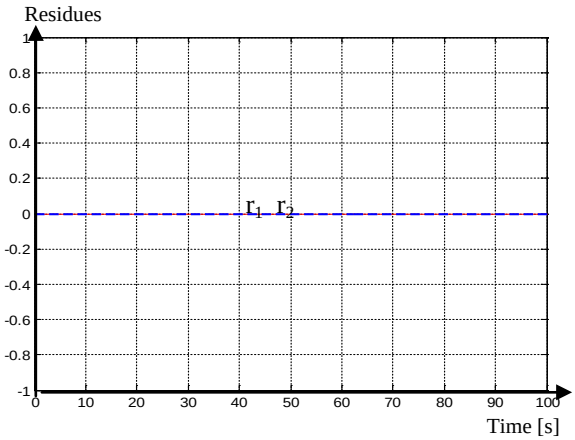

Fig. 4 The residues r_1 and r_2 in normal condition to the change of the prescribed size

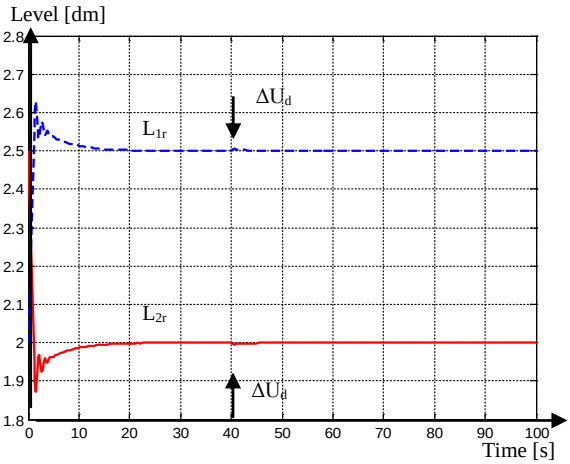

Fig. 5 The level L_{1r} and L_{2r} for ΔU_d=0.1

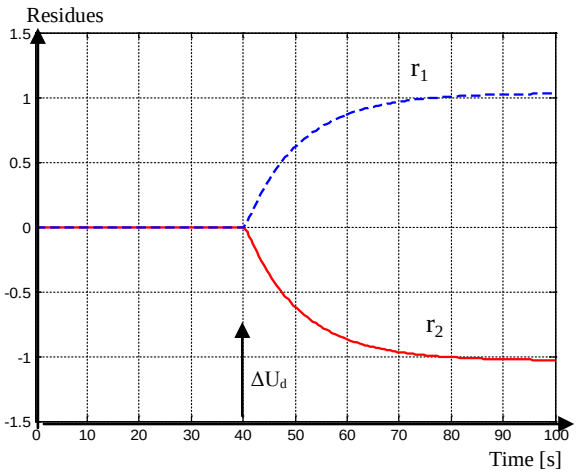

Fig. 6 The residues r_1 and r_2 for ΔU_d=0.1

In Fig 5and Fig. 6 are represented the level for the real plant, respective the residues in fault condition, were the residue is ΔU_d=0.1, applied at time t = 40 s.

In Fig 7and Fig. 8 are represented the level for the real plant, respective the residues in fault condition, when ΔU_d=0.5, applied at time t = 40 s.

In both case the residues are indicate the fault.

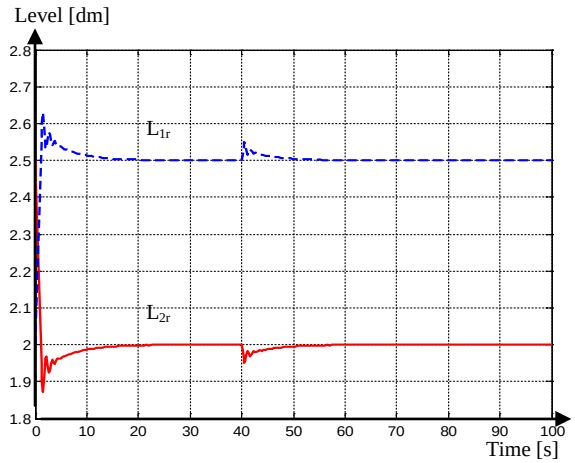

Fig. 7 The level L_{1r} and L_{2r} for ΔU_d=0.5

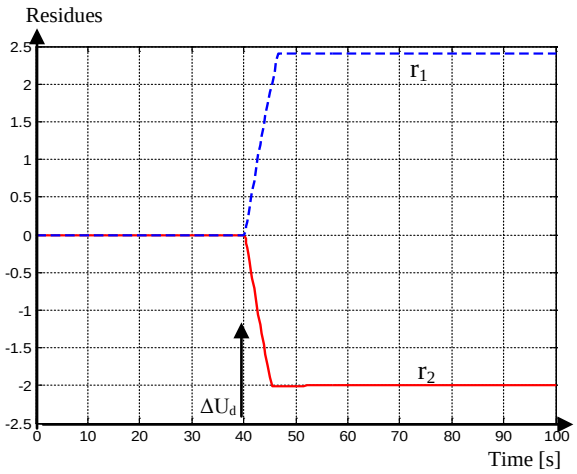

Fig. 8 The residues r_1 and r_2 for ΔU_d=0.5

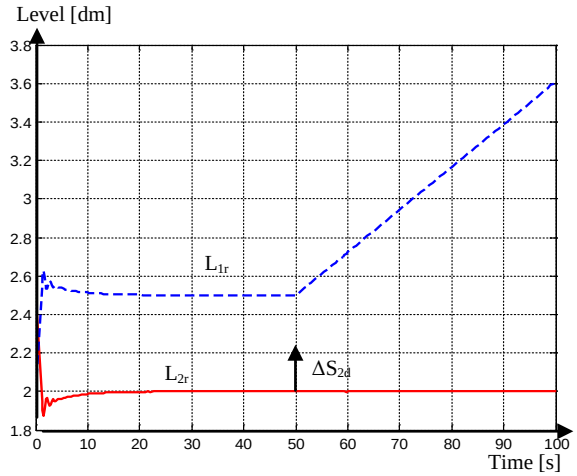

Fig. 9 The level L_{1r} and L_{2r} for ΔS_{2d}=0.05

Fig 9 and Fig. 10 represented the level for the real plant, respective the residues in fault condition, when ΔS_{2d}=0.05, applied at time t = 50 s. The new residue r_3 indicate this fault and the fault value.

Fig 11 and Fig. 12 represented the level for the real plant, respective the residues in fault condition, when ΔS_{2d}=0.1, applied at time t = 70 s.

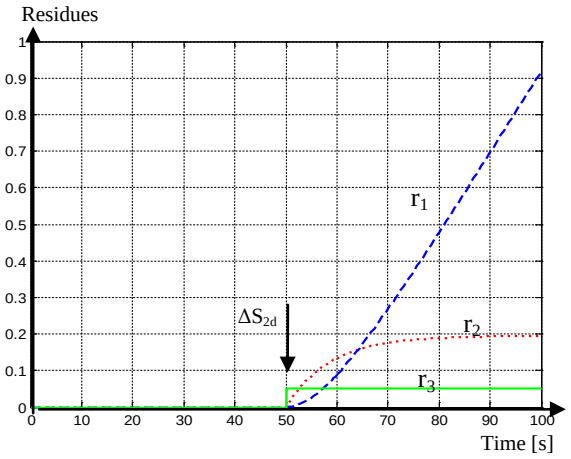

Fig. 10 The residues r_1, r_2 and r_3 for ΔS_{2d}=0.05

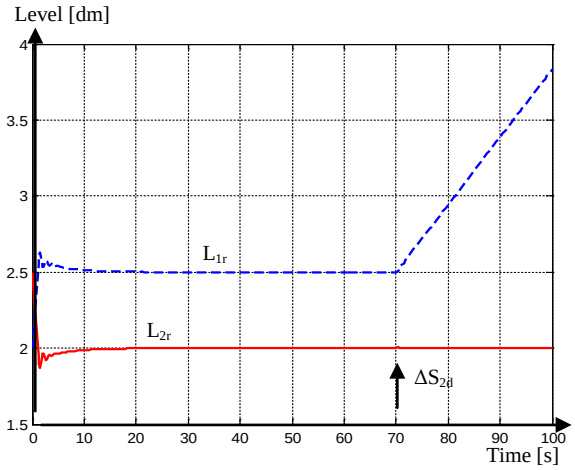

Fig. 11 The level L_{1r} and L_{2r} for ΔS_{2d}=0.1

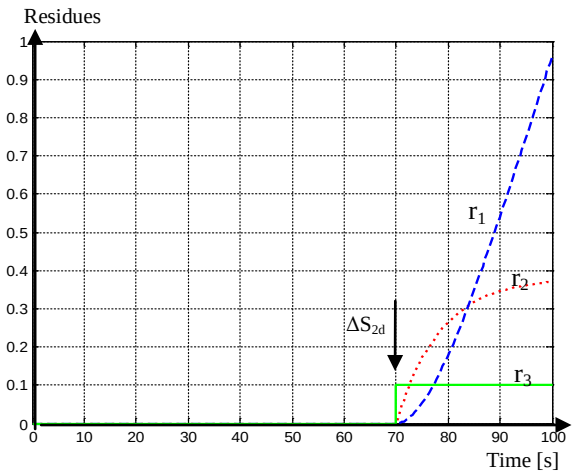

Fig. 12 The residues r_1, r_2 and r_3 for ΔS_{2d}=0.1

In Fig. 13 is presented the response of the real system, using the variables, L_{2r}- level in the tank2, which is changed as a result of the command given by the controller corresponding to the control loop of L_{2r}, so as to ensure the flow between L_1 and L_2 equal to the pump flow.

We notice that both of defects do not produce variations to the controlled value L_{2r}, them being compensated by the control system. To the L_{1r} variable can be determined only the fault ΔS_{2d}, applied at time t = 70 s, by its variation. So just based on variations of L_{1r} and L_{2r} cannot detect the defects, at most we can interpret the variation of L_{1r} if there is prior knowledge about the influence of ΔS_{2d} upon L_{1r}.

Instead the variations of the residues *r1, r2* and *r3* defined above, clearly highlights these defects. However, the defect ΔU_d strongly influences both the residues r_1 and r_2, but ΔS_{2d} has influence on residue r_3.

So we cannot locate both faults using residues r1 and r2. In this case, the residue r_3 is defined as:

$$r_3(s) = W(s)\begin{bmatrix} r_1(s) \\ r_2(s) \end{bmatrix} = \begin{bmatrix} W_{11}(s) & W_{12}(s) \end{bmatrix}\begin{bmatrix} r_1(s) \\ r_2(s) \end{bmatrix} = Z(s)\Delta S_{2d}(s)$$

From the transfer matrix $G_D(s)$ in the steady state resulting $W_{11} = W_{12} = 1$ and $r_3(s)$ is influenced only by the fault ΔS_{2d}. This is graphically represented in Fig.14.

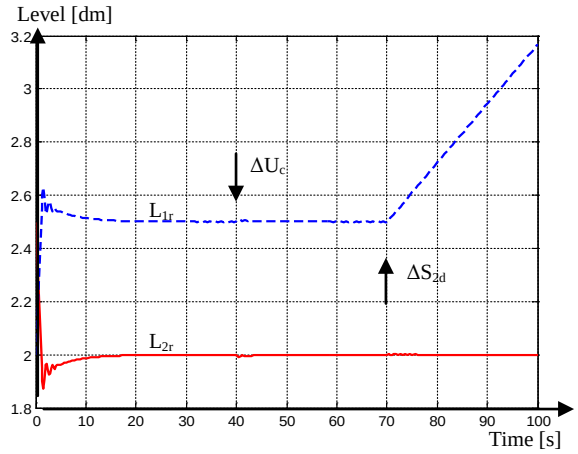

Fig. 13 The level L_{1r} and L_{2r} for ΔU_d=0.1 and ΔS_{2d}=0.05

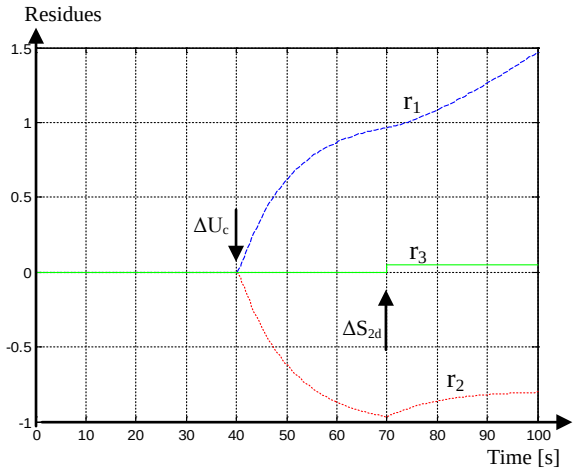

Fig. 14 The residues r_1, r_2 and r_3 for ΔU_d=0.1 and ΔS_{2d}=0.05

4. CONCLUSIONS

In this paper a method for faults detection and localization using residual vectors was presented; the proposed method was theoretically developed and experimentally verified. It allowed detection and localization of two faults created in a real process. A series of case studies was realized regarding the possibilities for detection of the faults using residuals for the system.

With the fault detection and localization scheme presented in this paper it can be detected the fault ΔU_d by the residue r_1 and the fault ΔS_{2d} by the residue r_3.

The advantage of this detection structure is that residue r_3 show us the fault value.

ACKNOWLEDGMENT

This work was supported by the strategic grant POSDRU/159/1.5/S/133255, Project ID 133255 (2014), co-financed by the European Social Fund within the Sectorial Operational Program Human Resources Development 2007 - 2013.

REFERENCES

Canureci, G., Vinatoru, M., and Maican C. (2012). *Fault detection and localization in dynamical systems*, Ed. SITECH, Craiova

Gertler, J., Staroswiecki M., and Shen, M. (2002). Direct Design of Structured Residuals for Fault Diagnosis in Linear Systems, *American Control Conference*, Anchorage, Alaska.

Iancu E. and Vinatoru M. (2003). *Analytical method for fault detection and isolation in dynamic systems study case*, Ed. Universitaria Craiova.

Korbicz, J., Koscielny J. M., Kowalczuk Z., and Cholewa W. (2004). *Fault diagnosis*, Ed. Springer.

Vinatoru, M. (2001). *Automatic control of the industrial process*, Ed. Universitaria, Craiova.

Vinatoru, M., Canureci, G., Maican C., and Iancu, E. (2007). Level and Temperature Control Study Using Festo Kit in a Laboratory Stand, *Proceedings of the 3rd WSEAS/IASME International Conference on Dynamical Systems and Control (CONTROL'07)*, Arcachon, Franta, pg. 247-252.

Vinatoru, M., Iancu, E., Canureci G., and Maican, C. (2008). *Automatic control of the industrial process - Design wizard and laboratory*, vol. 2, Ed. Universitaria Craiova.

ANNALS OF THE UNIVERSITY OF CRAIOVA
Series: Automation, Computers, Electronics and Mechatronics, Vol. 12 (39), No. 1, 2015

Compact Dynamic Model of the Brushless DC motor

Sergiu Ivanov*, Dan Selişteanu,**
Virginia Ivanov*, Dorin Şendrescu**

** Faculty of Electrical Engineering, University of Craiova, Romania*
(e-mail: sergiu.ivanov@ie.ucv.ro, vivanov@elth.ucv.ro).
*** Faculty of Automation, Computer and Electronics, University of Craiova, Romania*
(e-mail:{dansel, dorins}@automation.ucv.ro)

Abstract: Both the researchers and manufacturers are interested by the light urban electric vehicles. More ways are investigated for the used motors, the most important concern being the compactness of the solution. Thanks to the high power density, the most preferred motors are the permanent magnet synchronous motors (PMSM) and brushless DC motors (BLDC). These are preferred also thanks to their high efficiency and low maintenance cost. For both, the main manufacturing technology and associated power electronics are quite similar. The differences occur in the controlling technology, for BLDC being simpler and more advantageous. As integration technology, the direct drive in-wheel technology improves the safety, efficiency, weight, controllability and finally the costs. The development of the control strategies implies the need for a simple and compact model of the BLDC. The paper deals with an efficient dynamic model of this type of motor. It will be applied for the most used control strategy, the preset currents respectively.

Keywords: BLDC motor model, S-function, in-wheel motor, control.

1. INTRODUCTION

The brushless DC (BLDC) motors become in the last years quite interesting for different industrial and home applications (servo drives, peripherals, small vehicles), thanks to a series of advantages related to the small volume, high power density (light-weight), efficiency (Yedamale, 2003).

More communications report the use of such type of motor for traction purposes (Tashakori et al., 2011). The direct drive in-wheel technology is preferred more and more for low scale traction applications. The inclusion of separated motors in each wheel eliminates the need of a central drive and the corresponding complicated, imprecise and heavy mechanisms necessary for distribute the torque to the wheels (Tashakori et al., 2010).

The BLDC motor is quite similar as construction with the permanent magnets synchronous one. Consequently, the specified advantages are common. The difference occurs in the control technique. For the BLDC is much simpler, as it needs only the identification of the one of the six 60º sectors where the rotor is situated. Contrary, the PMSM needs an absolute, precise position encoder.

Several publications (Rambabu, 2007; Prasad et al., 2012; Fedák et al., 2012, Singh et al., 2013) deeply describe the mathematical model of the BLDC. The practical implementation depends on the researchers experience and technical options.

Concerning the command, the natural or phase variable model offers many advantages thanks to the trapezoidal back EMF, which avoids the well-known Park transformation necessary when sinusoidal variation of the motor inductances with rotor angle occur, as is the case of the PMSM.

The paper presents an original and very compact implementation of the BLDC model by using the S-function facility of Simulink. The most used control strategy is implemented by using the developed motor model, confirming its viability.

2. BLDC MOTOR MODEL

By considering several hypothesis, the phase variable model is implemented: the stator phase resistances are constant (is neglected the temperature influence), all the inductances are constant (magnetic saturation is not present), hysterezis and eddy current losses are not considered and all the inverter switches are ideal.

With the above assumptions, the dynamic model of the BLDC motor is described by the voltage equations on the three phases:

$$u_a = R_s i_a + \left(L_s - L_m\right)\frac{di_a}{dt} + e_a, \tag{1}$$

$$u_b = R_s i_b + \left(L_s - L_m\right)\frac{di_b}{dt} + e_b, \tag{2}$$

$$u_c = R_s i_c + \left(L_s - L_m\right)\frac{di_c}{dt} + e_c. \tag{3}$$

In order to obtain the dynamic model in the state variables form, it is advantageous to write (1-3) in matrix form

$$[u]=[R][i]+[L_s-L_m]\frac{d}{dt}[i]+[e], \quad (4)$$

where

$[u]=[u_a \quad u_b \quad u_c]^T$ is the input voltages vector;

$[R]=\begin{bmatrix} R_s & 0 & 0 \\ 0 & R_s & 0 \\ 0 & 0 & R_s \end{bmatrix}$ is the stator resistances diag. matrix;

$[i]=[i_a \quad i_b \quad i_c]^T$ is the phases currents vector;

$[L_s-L_m]=\begin{bmatrix} L_s-L_m & 0 & 0 \\ 0 & L_s-L_m & 0 \\ 0 & 0 & L_s-L_m \end{bmatrix}$ is the stator inductances diagonal. matrix;

$[e]=[e_a \quad e_b \quad e_c]^T$ is the back emf vector.

With this matrix form, the state space model is simply obtained:

$$[\dot{i}]=\frac{d}{dt}[i]=[L_s-L_m]^{-1}([u]-[R][i]-[e]) \quad (5)$$

Taking into account the diagonal form of the inductances matrix, its inverse has as elements, the inverse of each element.

The back emf vector [*e*] has as elements:

$$e_a=\omega\cdot K_e\cdot f(\theta_e), \quad (6)$$

$$e_b=\omega\cdot K_e\cdot f(\theta_e-2\pi/3), \quad (7)$$

$$e_c=\omega\cdot K_e\cdot f(\theta_e-4\pi/3), \quad (8)$$

where θ_e is the electric angle of the rotor, ω is the angular speed of the rotor, K_e back emf constant [V/(rad/sec)] and the reference function $f(\theta_e)$ of the back emf is alternative, with trapezoidal shape and amplitude equal to 1. In Fig. 1 are plotted the "*a*" phase waveforms of the reference function fa(θe), preset current ia* and the Hall signal.

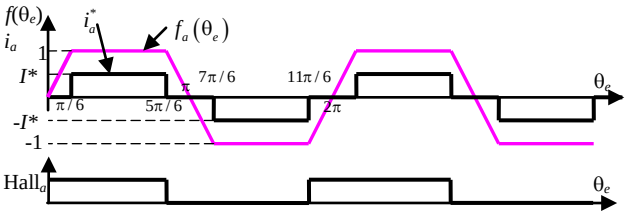

Fig. 1. The waveforms for the phase *a*.

For the other two phases, all the signal are delayed by 2π/3 and 4π/3 respectively:

$$f_b(\theta_e)=f_a\left(\theta_e-\frac{2\pi}{3}\right), \quad i_b^*=i_a^*\left(\theta_e-\frac{2\pi}{3}\right), \quad \text{Hall}_b=\text{Hall}_a\left(\theta_e-\frac{2\pi}{3}\right),$$

$$f_c(\theta_e)=f_a\left(\theta_e-\frac{4\pi}{3}\right), \quad i_c^*=i_a^*\left(\theta_e-\frac{4\pi}{3}\right), \quad \text{Hall}_c=\text{Hall}_a\left(\theta_e-\frac{4\pi}{3}\right).$$

The model described by (5) is completed with the torque expression

$$m=\frac{e_a i_a+e_b i_b+e_c i_c}{\omega}= \\ =K_e\cdot\left[i_a\cdot f_a(\theta_e)+i_b\cdot f_b(\theta_e)+i_c\cdot f_c(\theta_e)\right]. \quad (9)$$

and with the movement equation,

$$m=J\cdot\frac{d\omega}{dt}+B\cdot\omega+m_s. \quad (10)$$

where m and m_s are the electromagnetic torque and the static one respectively, J is the total inertia and B is the viscous frictions coefficient.

From (10) it results the fourth state equation:

$$\dot{\omega}=\frac{d\omega}{dt}=\frac{1}{J}(m-m_s-B\cdot\omega). \quad (11)$$

The state variables will be the phase currents and the speed.

The link between the electric angle θ_e and the mechanical one is done by the number of pairs of poles

$$\theta_e=p\cdot\theta_m. \quad (12)$$

The mathematical model described by (5-9, 11-12) was implemented by using the S-function facility of Simulink. Fig. 2 depicts the result.

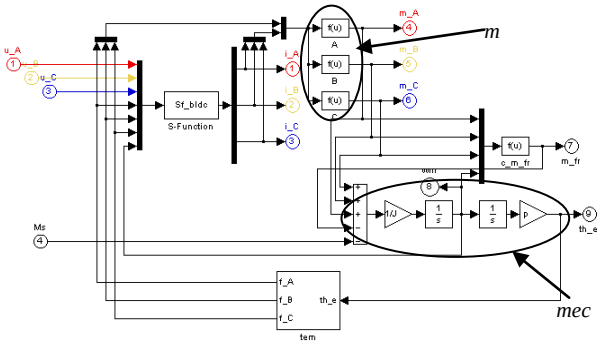

Fig. 2: The Simulink model of the BLDC with S-function.

The highlighted areas correspond to the computation of the electromagnetic torque *m*, by using (9) and to the integration of the mechanical state equation (11).

All the dynamic model described by (5) is written in an *m*-file (Fig. 3) which is called with different flags by the S-function block during the simulation. In the zone 1, the inputs (phase voltages) are updated. In the zone 2, the state derivatives are computed with (5). In zone 3 the outputs (phase currents) are computed.

The block tem from Fig. 2 generates, based on the electric angle reduced to the 0-2π range, the reference functions of the back emf (6-8) shown in Fig. 4, used both for integration of the dynamic model (5) and for electromagnetic torque computation (9).

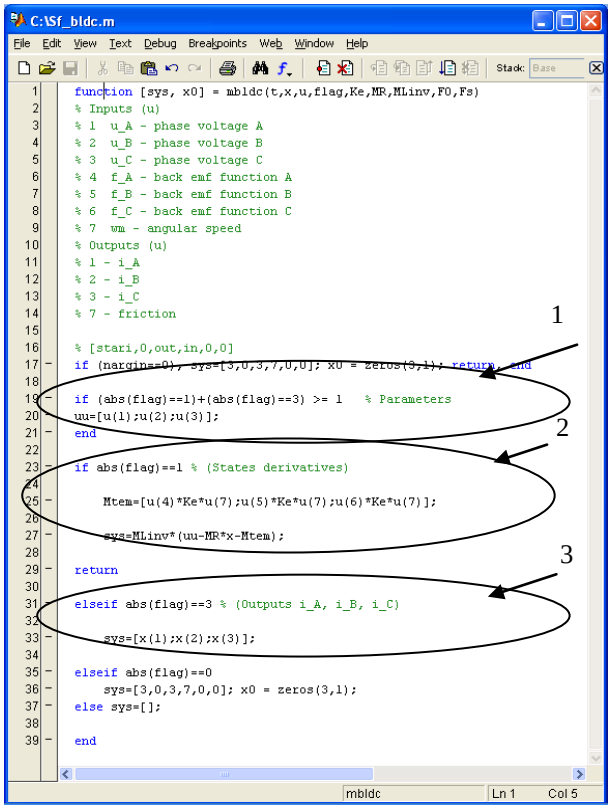

Fig. 3: The S-function corresponding to the BLDC motor.

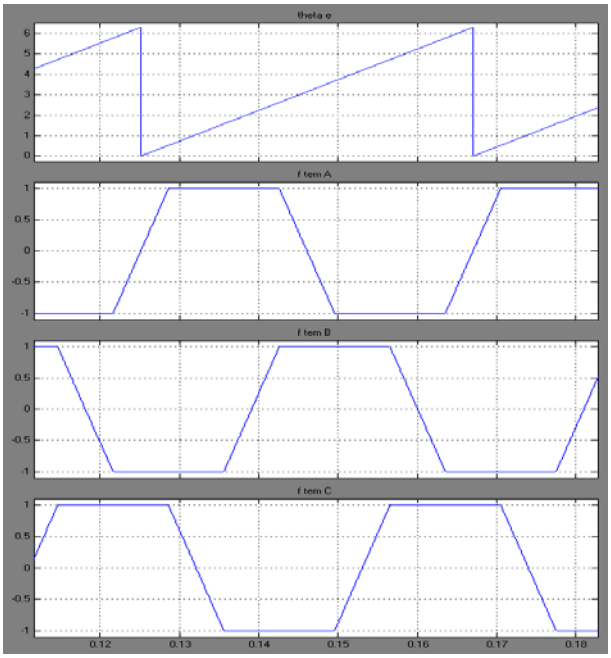

Fig. 4: The reference back emf functions f_a, f_b, f_c.

As can be seen, the Simulink model is quite intuitive and compact. The values of all the parameters are transferred by the dialog box (Fig. 5) of the grouped model shown in Fig. 2.

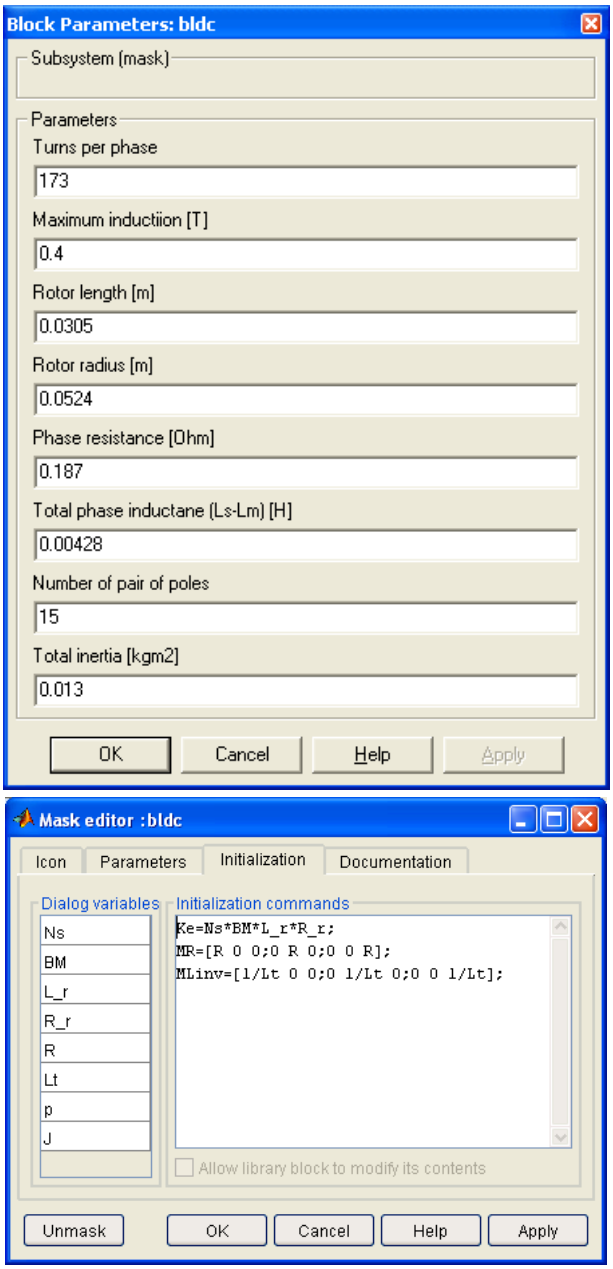

Fig. 5: Dialog box of the BLDC motor model

3. COMMAND AND SIMULATIONS

Basically, the command grants the classical 120º square currents, in phase with the back emf, in order to maximize the developed torque.

The experienced method is based on the preset current modulation, which determines that the inverter is a current source.

The principle of this type of command is depicted in Fig. 6.

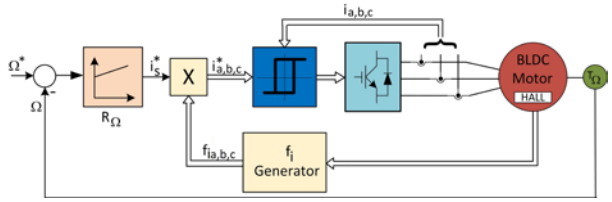

Fig. 6: Principle diagram for the preset currents command

Based on the information delivered by Hall sensors on the motor shaft, the block "f_i Generator" generates, the 120º square waveforms of the currents in p.u. The amplitude of the currents is obtained as result of the PI speed controller which multiplies the unitary waveforms in order to obtain the preset stator currents. These are compared then with the real ones by using three hysterezis comparators, like in the well-known bang-bang modulation. The result pulses are applied to the six switches of the inverter bridge.

Fig. 7 plots few results of the simulation: step start at no load (0.5 Nm) and 10 rad/sec, followed by a load application (1.5 Nm) at 0.14 s and an acceleration to 15 rad/sec with the applied load.

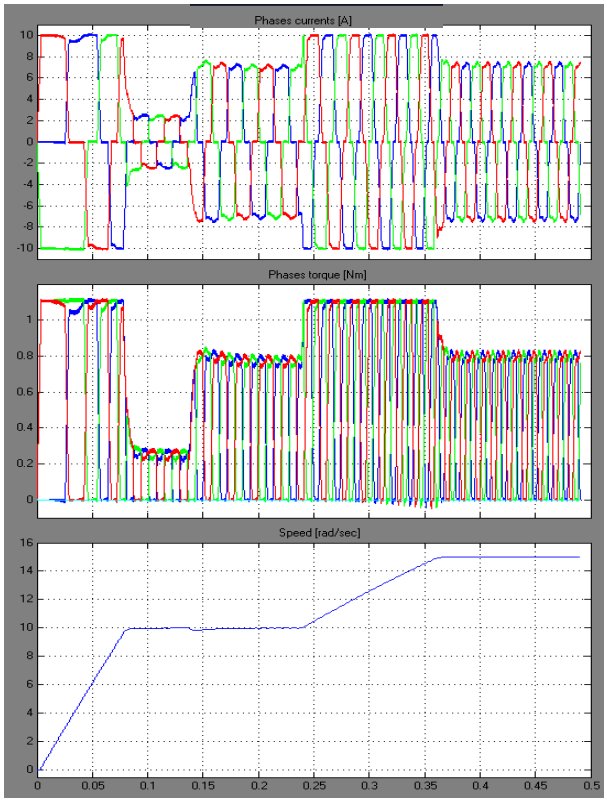

Fig. 7: Results for preset currents command

We notice the very good dynamic behaviour, the command and waveforms accuracy. Only the motor inertia was considered.

The command method has the advantage to be quite simple to be implemented, as low cost analogue hysterezis controllers can be used. Also, the method is not sensitive to the variations of the motors parameters, especially stator resistance and to the variations of the supplying voltage. The main disadvantages are related to the variable switching frequency and application size, as there are necessary fast switching elements, available only at low to medium power.

4. CONCLUSIONS

The paper presents a compact and simple implementation of the BLDC motor dynamic model, by using the S-function facility of Simulink. One type of command is implemented in order to test the model viability. The advantages and disadvantages are emphasized and possible applications highlighted. The future activity will be focused on the development of different commands types.

ACKNOWLEDGMENT

This work was partially supported by the grant number P09004/1137/31.03.2014, cod SMIS 50140, entitled: Industrial research and experimental development vehicles driven by brushless electrical motors supplied by lithium-ion accumulators for people transport - GENTLE ELECTRIC.

REFERENCES

Fedák, V., Balogh, T., and P. Záskalický. (2012). *Dynamic Simulation of Electrical Machines and Drive Systems Using MATLAB GUI- A Fundamental Tool for Scientific Computing and Engineering Applications* - Volume 1, Prof. Vasilios Katsikis (Ed.), ISBN: 978-953-51-0750-7, InTech.

Prasad, G., Sree Ramya, N., Prasad, P.V.N., and Tulasi Ram Das G. (2012). Modelling and Simulation Analysis of the Brushless DC Motor by using MATLAB, *International Journal of Innovative Technology and Exploring Engineering*, Vol. 1, Issue 5.

Rambabu, S. (2007). *Modeling and control of a brushless DC motor*, PhD Thesis.

Singh, C.P., Kulkarni, S.S., Rana, S.C., and Kapil D. (2013). State-Space Based Simulink Modeling of BLDC Motor and its Speed Control using Fuzzy PID Controller. *International Journal of Advances in Engineering Science and Technology*, Vol. 2 , No. 3, pp. 359-369.

Tashakori, A., Ektesabi, M., and Hosseinzadeh, N. (2010). Characteristic of suitable drive train for electric vehicle. In *Proceedings of the 3rd International Conference on Power Electronic and Intelligent Transportation System* (PEITS), 20-21 Nov. 2010, Shenzhen, China.

Tashakori, A., Ektesabi, M., and Hosseinzadeh, N. (2011). Modeling of BLDC Motor with Ideal Back-EMF for Automotive Applications. In *Proceedings of the World Congress on Engineering, WCE 2011*, July 6-8, 2011, London, U.K.

Yedamale, P. (3003). *Brushless DC (BLDC) Motor Fundementals*. AN885, 2003. Michrochip Technology Inc.

ANNALS OF THE UNIVERSITY OF CRAIOVA
Series: Automation, Computers, Electronics and Mechatronics, Vol. 12 (39), No. 1, 2015

Adaptive and Predictive Control Algorithms for a Microalgae Process

Emil Petre

Department of Automatic Control and Electronics, University of Craiova, Craiova, Romania
(e-mail: epetre@ automation.ucv.ro)

Abstract: This paper deals with the design and the analysis of two control structures for microalgae culture process to regulate the substrate concentration at a chosen setpoint. The control strategies are developed under the realistic assumptions that the reaction rates are time varying and incompletely known and the influent substrate concentration as well as the light intensity are strongly time-varying. The first control structure is an adaptive control algorithm. This controller is designed by coupling a linearizing controller with a parameter estimator used for estimation of unknown kinetics. The predictive control structure is based on classical nonlinear model predictive control law under model parameter uncertainties implying solving a nonlinear least squares optimization problem for setpoint trajectory tracking. The proposed approaches are validated in simulation and numerical results are given to illustrate its efficiency for setpoint tracking in the presence of parameters uncertainties.

Keywords: Bioprocesses, Microalgae, Droop model, Photobioreactor, Adaptive control, Predictive control, Nonlinear least squares optimization.

1. INTRODUCTION

It is well known that water is an essential element of life being an important resource both for industrial applications and domestic usage. Therefore in last time numerous environmental laws and directives have arisen in order to decrease the pollution related to the industrial and urban effluents. This situation has led to an increase in the use of wastewater biological treatment processes. For example, the anaerobic digestion is very useful since it produces valuable energy (methane) besides removing the organic pollution from the liquid influent (Angulo et al., 2007). Nevertheless, its main drawback is the production of carbon dioxide (CO_2) and its easy destabilization (Angulo et al., 2007; Bastin and Dochain, 1990; Bernard, 2004). Therefore the researchers have been searched solutions to improve the efficiency in the pollution reduction and for CO_2 mitigation (Bernard, 2011; Benattia et al., 2014a, b; Ifrim et al., 2013). A recently used solution consist in the growth of some microalgae populations (in fact, autotrophic microalgae and cyanobacteria (Bernard, 2011) that by using light as source of energy are able to assimilate inorganic forms of carbon (CO_2, HCO_3^-) and to convert them into requisite organic substances intended to many industrial applications: food, pharmacology, chemistry, generating at the same time oxygen (O_2) (Bernard, 2011; Benattia et al., 2014a, b; Ifrim et al., 2013). Microalgal biofuel production systems could also contribute to mitigate industrial CO_2 emitted from power plants, cement plants, etc. In the same spirit, microalgae could be used to consume inorganic nitrogen and phosphorus in urban or industrial effluents, and thus limit expensive wastewater post-treatment (Bernard, 2011).

However, microalgae have been so far only marginally used for biotechnological applications as: vitamins, proteins, cosmetics, and health foods. But in perspective of large scale microalgal cultivation, the modelling and control of such processes remains a key issue for the enhancement of stability and process efficiency since microalgae have some specificities compared to microorganisms such as bacteria or yeasts (Bernard, 2011). A difficulty for the design of high-performance control techniques of such living processes lies in the fact that, in many cases, the models contain kinetic parameters and/or yield coefficients that are highly uncertain and time varying (Bastin and Dochain, 1990; Batstone et al., 2002; Bernard, 2011; Dochain and Vanrolleghem, 2001; Mairet et al., 2011). Therefore most of them must to be estimated (Bastin and Dochain, 1990; Dochain and Vanrolleghem, 2001).

To control these processes, several control strategies were developed such as linearizing feedback (Angulo et al., 2007; Bastin and Dochain, 1990; Dochain, 2008; Neria-González et al., 2009; Petre et al., 2013; Petre and Selişteanu, 2013; Tebbani et al., 2014a), adaptive and robust-adaptive approach (Bastin and Dochain, 1990; Neria-González et al., 2009; Petre et al., 2013; Petre and Selişteanu, 2013), predictive an optimal control (Bernard, 2011; Benattia et al., 2014a, b; Logist et al., 2011; Tebbani et al., 2014, 2015), sliding mode (Selişteanu et al., 2007), neural techniques (Hayakawa et al., 2008; Petre et al., 2010), and so on. Some of these approaches imposed the use of the so-called "software sensors", used not only for the estimation of concentrations of components but also for the estimation of kinetic parameters or even reaction rates (Bastin and Dochain, 1990; Dochain and Vanrolleghem, 2001; Neria-González et al., 2009; Petre et al., 2013; Petre and Selişteanu, 2013).

The aim of this paper is to develop some adaptive and predictive control structures, able to deal with the model uncertainties, for a microalgae culture process that is carried out in a continuous perfectly mixed bioreactor. The control algorithms are developed under the realistic assumptions that the reaction rates are time varying and incompletely known and the influent substrate

concentration as well as the light intensity are strongly time-varying. The adaptive control structure is achieved by combining a linearizing control law with a parameter estimator which plays the role of a software sensor for on-line estimation of unknown bioprocess kinetic rates.

The predictive control structure is based on Nonlinear Model Predictive Control (NMPC) strategy (Camacho and Bordons, 2004). The main advantage of NMPC law is that it allows the current control input to be optimized, while taking into account the future system behaviour. This is achieved by optimizing the control profile over a finite time horizon, but applying only the current control input (Benattia et al., 2014a, b).

The behaviour and performance of the proposed control algorithms are illustrated by numerical simulations applied in the case of a continuous microalgae bioprocess for which the bacterial growth rate is strongly nonlinear and time varying, the uptake rate is completely unknown, and the influent substrate concentration as well as the light intensity are strongly time-varying.

2. PROCESS DESCRIPTION AND MODELLING

The well-known model of microalgal growth is the Droop model. This model takes into account the specificity of microalgae in comparison to other microorganisms that is they use light to grow and inorganic substrate uptake and growth are decoupled thanks to an intracellular storage of nutrients (Bernard, 2011; Benattia et al., 2014a, b). The Droop model involves three state variables: the biomass concentration X, in $\mu m^3\ L^{-1}$, the limiting substrate (dissolved inorganic nitrogen (nitrate or ammonium)) concentration S, in $\mu mol\ L^{-1}$, and the internal quota Q, in $\mu mol\ \mu m^{-3}$, defined as the quantity of substrate per unit of biomass (Bernard, 2011; Benattia et al., 2014a, b). The considered dynamic model assumes that the process takes place in a perfectly mixed continuous photobioreactor (the influent flow rate equals the effluent flow rate, leading to a constant effective volume), without any additional biomass in the feed, and neglecting the effect of gas exchanges. Then the differential equations which describe this process (resulting from mass balances) are given by (Benattia et al., 2014a, b):

$$\begin{aligned} \dot{X}(t) &= \mu(Q(t), I(t))X(t) - DX(t) \\ \dot{Q}(t) &= \rho(S(t)) - \mu(Q(t), I(t))Q(t) \\ \dot{S}(t) &= \rho(S(t))X(t) + D(S_{in} - S(t)) \end{aligned} \quad (1)$$

where D is the dilution rate (d^{-1}, d: day) and S_{in} the influent substrate (inorganic nitrogen) concentration ($\mu mol\ L^{-1}$).

The specific uptake rate $\rho(S)$ is represented by a Monod model (Benattia et al., 2014a, b):

$$\rho(S) = \rho_m \frac{S(t)}{K_s + S(t)}, \quad (2)$$

where ρ_m and K_s represent respectively the maximal specific uptake rate and the substrate half saturation constant.

The specific growth rate $\mu(Q, I)$ is modelled as (Benattia et al., 2014a, b):

$$\mu(Q,I) = \overline{\mu}\left(1 - \frac{K_Q}{Q(t)}\right)\mu_I(I(t)), \quad (3)$$

where $\overline{\mu}$ is the theoretical specific maximal growth rate, i.e. the growth rate at hypothetical infinite quota (Bernard, 2011), K_Q represents the minimal cell quota allowing growth, and $\mu_I(I)$ denotes the light effect. This is modelled by a Haldane model which describes the photo-inhibition phenomenon as (Benattia et al., 2014a, b):

$$\mu_I(I) = \frac{I}{I + K_{sI} + I^2 / K_{iI}}, \quad (4)$$

where I is the light intensity (in $\mu E\ m^{-2}\ s^{-1}$) and K_{sI} and K_{iI} are light saturation and inhibition constants respectively. The optimal light intensity that maximises the function $\mu_I(I)$ is given by $I_{opt} = \sqrt{K_{sI} K_{iI}}$ (Benattia et al., 2014a). In the sequel, we will consider that the light intensity value is not set at this optimal value I_{opt}; it is assumed a strongly time varying function (that should simulate some cloudy or rainy days and nights).

The meaning and the values parameters in the Droop model are given in the Table 1 (Benattia et al., 2014b).

Table 1. Parameter values in the Droop model.

Parameter	Value and unit	Meaning
ρ_m	9.3 $\mu mol\ \mu m^{-3}\ d^{-1}$	Maximal specific uptake rate
K_s	0.105 $\mu mol\ L^{-1}$	Substrate half saturation constant
$\overline{\mu}$	2 d^{-1}	Theoretical specific maximal growth rate
K_Q	1.8 $\mu mol\ \mu m^{-3}$	Minimal cell quota
K_{sI}	150 $\mu E\ m^{-2}\ s^{-1}$	Light saturation constant
K_{iI}	2000 $\mu E\ m^{-2}\ s^{-1}$	Light inhibition constant
S_{in}	100 $\mu mol\ L^{-1}$	Microalgae substrate concentration

3. CONTROL STRATEGIES

For the microalgae process described by the dynamical model (1)-(4) we consider the control problem of the internal substrate concentration at a chosen setpoint by using as control input the dilution rate D, the aim being also the obtaining of a great quantity of biomass. The control problem will be resolved under some realistic conditions specified in the previous section.

3.1. Exact linearizing control law

Firstly, we consider the case when the whole bioprocess is completely known (the ideal case). This means that the model (1)-(4) is completely known (i.e. all the specific rates are assumed completely known, and all the state variables and the inflow rate are available by on-line measurements). Since in model (1) the dynamics of S has the relative degree equal to 1, then the following exact linearizing control law:

$$D(t) = 1/(S_{in} - S) \cdot \left(\dot{S}^* + \lambda(S^* - S) + \rho(S)X\right), \quad (4)$$

where S^* denotes the desired set point assures a stable behaviour of closed-loop system described by the following first order linear stable differential equation:

$$(\dot{S}^* - \dot{S}) + \lambda(S^* - S) = 0,\ \lambda > 0. \quad (5)$$

where λ is a design parameter. The control law (4) leads to a linear dynamics of the tracking error $e = y^* - y$ described by $\dot{e} = -\lambda e$, which for $\lambda > 0$ has an exponential stable point at $e = 0$. This control law will be used in order to design the adaptive algorithm as well as benchmark in order to compare the behaviour of the closed loop system in this case with the behaviour of systems controlled by using the proposed adaptive and predictive algorithms.

3.2. An adaptive control algorithm

Since the prior knowledge about the bioprocess supposed before is not realistic, in the following we develop an adaptive control strategy under the following conditions:

- the specific uptake rate ρ is incompletely known, since ρ_m and K_s are considered unknown;
- the state variable X is not measurable;
- the on-line available measurements are S and S_{in}, but S_{in} is time varying;
- the light intensity value is not set at this optimal value I_{opt} and it is considered a strongly time varying function;
- the internal quota Q can be calculated, but this is not used in the control design;
- all the other kinetic and process coefficients are known.

Under these conditions an adaptive controller is obtained as follows. Since X is not measurable, and ρ is incompletely known, then in control law (4) the whole uptake rate $\rho(S)X = \varphi_u$ will be considered an unknown function that will be estimated by using an appropriately parameter estimator. Here we will use an observer-based parameter estimator (OBE) (for details see (Bernard, 2011; Dochain and Vanrolleghem, 2001; Petre et al., 2013; Petre and Selişteanu, 2013)). Since for this process we must to estimate only one completely unknown reaction rate, then using the dynamics of S, the proposed OBE is described by the following equations:

$$\begin{aligned} \dot{\hat{S}} &= -\hat{\varphi}_u + D(S_{in} - S) - \omega(\hat{S} - S), \\ \dot{\hat{\varphi}}_u &= -\gamma(\hat{S} - S), \end{aligned} \quad (6)$$

where $\hat{\varphi}_u$ is the on-line estimate of the unknown φ_u, and $\omega < 0$ and $\gamma > 0$ are tuning parameters to control the stability and the tracking properties of the estimator. Usually, the values of these parameters are choosing by trial and error method.

Then, the proposed adaptive control algorithm is obtained by combination of the parameter estimator equations (41)-(46) with the control law (26) rewritten as:

$$D(t) = 1/(S_{in} - S)\cdot\left(\dot{S}^* + \lambda(S^* - S) + \hat{\varphi}_u\right). \quad (7)$$

A block diagram of the proposed adaptive control system is shown in Fig. 1.

3.3. A nonlinear model predictive control

Now, a nonlinear model predictive control (NMPC) problem will be formulated for general case as follows.

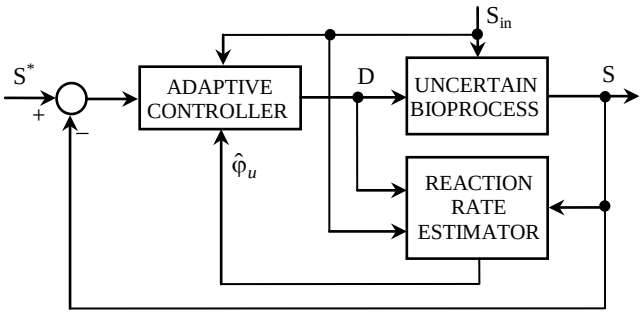

Fig. 1. Scheme of the adaptive closed-loop controlled system

Consider the following discrete-time, time-invariant nonlinear system:

$$\begin{cases} \xi_{k+1} = f(\xi_k, u_k) \\ y_k = h(\xi_k, u_k) \end{cases} \quad (8)$$

where ξ_k is the state vector, u_k is the control input, y_k is the process output, and f and h are nonlinear smooth functions. The control objective is to regulate the output variable y_k to a specified setpoint value y_{ref} under certain state and input constraints:

$$\begin{aligned} \xi_{\min} \le \xi_k \le \xi_{\max}, \\ u_{\min} \le u_k \le u_{\max}. \end{aligned} \quad (9)$$

By using NMPC the solution of this constrained control problem is obtained by repeatedly solving the following optimization problem:

$$\min\left(\sum_{i=1}^{N}(y_{ref} - y_{k+i})^T \Psi(y_{ref} - y_{k+i}) + \sum_{i=1}^{N_u} u_{k+i}^T \Omega u_{k+i}\right) \quad (10)$$

so that

$$\begin{cases} \xi_{k+1} = f(\xi_k, u_k) \\ \xi_{\min} \le \xi_k \le \xi_{\max} \\ u_{\min} \le u_k \le u_{\max} \end{cases}, \quad (11)$$

where Ψ and Ω are two positive semidefinite matrices, N denotes the length of the prediction horizon and N_u the length of the control horizon.

The model predictive control is a strategy that is based on the explicit use of some kind of system model to predict the controlled variables over a certain time horizon, called the prediction horizon (Eaton et al., 1990; Mayne et al., 2000).

The NMPC control strategy used in this paper can be structured described as follows (Eaton et al., 1990):

1. A reference trajectory $y_{ref}(t+k)$, $k = 1,\ldots,N$ is defined which describes the desired system trajectory over the prediction horizon.
2. At each sampling time, the value of the controlled variable $y(t+k)$ is predicted over the prediction horizon $k = 1,\ldots,N$. This prediction depends on the future values of the control variable $u(t+k)$ within a control horizon $k = 1,\ldots,N_u$.

3. The vector of future controls $u(t+k)$ is computed such that an objective function (a function of the errors between the reference trajectory and the predicted output of the model) is minimised.

4. Once the minimisation is achieved, only the first optimised control action is applied to the plant and the plant outputs are measured. This measurement as well as the plant current state (either measured or estimated) is used as the initial states of the model to perform the next iteration (Șendrescu et al., 2011).

Steps 1 to 4 that are repeated at each sampling instant are called a receding horizon strategy.

The MPC based control strategy can be represented by the scheme shown in Fig. 2.

Since usually a solution of the nonlinear least squares (NLS) minimization problem cannot be obtained analytically then it is computed by using numerical methods. There are many different methods of numerical optimization. To solve the NLS optimisation problem it was chosen the Levenberg-Marquardt (LM) algorithm.

The LM algorithm is an iterative technique that locates the minimum of a multivariate function that is expressed as the sum of squares of non-linear real-valued functions (Kouvaritakis and Cannon, 2001; Wang and Boyd, 2008). It has become a standard technique for non-linear least-squares problems (Nocedal and Wright, 1999), widely adopted in a broad spectrum of disciplines. LM can be thought of as a combination of steepest descent and the Gauss-Newton method (Șendrescu et al., 2011). When the current solution is far from the correct one, the algorithm behaves like a steepest descent method. When the current solution is close to the correct solution, it becomes a Gauss-Newton method (Șendrescu et al., 2011).

The previous general NMPC formulation is applied to microalgae process, in order to regulate the internal substrate S to a reference value S^* by manipulating the dilution rate D under the same realistic conditions about the process as in the design of the adaptive control algorithm.

The implementation of the presented predictive control strategies requires a discrete-time model of the process (1)-(4). This can be obtained by using, for example, the Euler approximation.

4. SIMULATION RESULTS AND DISCUSIONS

The behaviour and performance of the proposed adaptive and predictive control algorithms are examined by using simulation experiments in the case of the presented microalgae process for which the bacterial growth rate is strongly nonlinear and time varying, the uptake rate is completely unknown, and the influent substrate concentration as well as the light intensity is strongly time-varying. The behaviour of these algorithms is compared to the exact linearizing control law (4) considered as benchmark. The simulations are achieved using the model (1)-(4) under identical circumstances. For the yield and kinetic coefficients are used the values given in the Table 1.

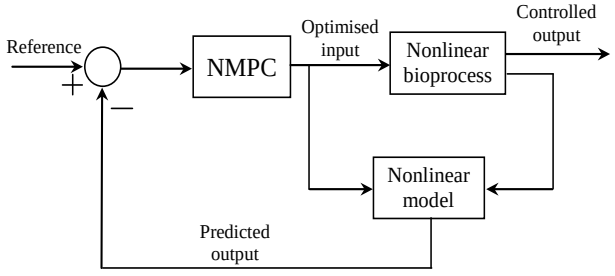

Fig. 2. NMPC control strategy

Case 1: Adaptive control. The performance of the closed-loop system with the adaptive controller (7) is analysed, by comparison to linearizing controller (4), under the next conditions:

- the specific uptake rate ρ is incompletely known, that is ρ_m and K_s are considered unknown;
- the state variable X is not measurable;
- the on-line available measurements are S and S_{in}, but S_{in} is time varying as in Fig. 3;
- the light intensity value is not set at this optimal value I_{opt}; it is assumed a strongly time varying function (that simulates some cloudy or rainy days and nights day), Fig. 4;
- Q is calculated, but this is not used in the control design;
- all the other kinetic and process coefficients are known.

The behaviour of microalgae process in closed-loop system under these conditions is presented in Fig. 5-9. The graphics in Fig. 5 correspond to the controlled variable S, and Fig. 6 plots the control input D. To verify the regulation properties of control laws, a piece-wise constant variation was taken into consideration for S^*.

From Fig. 4 and Fig. 5 it can be observed that the substrate concentration S tracks the reference profile S^*, and the control input D is maintained in the physical limits.

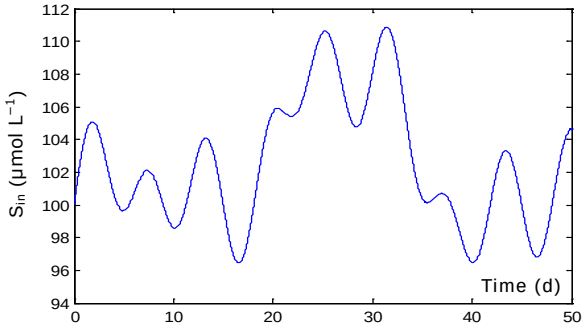

Fig. 3. Time evolution of the influent substrate S_{in}

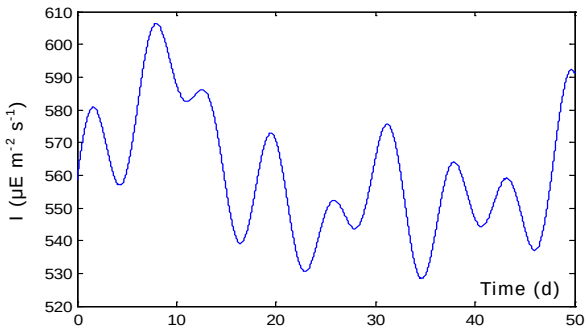

Fig. 4. Time evolution of the light intensity I

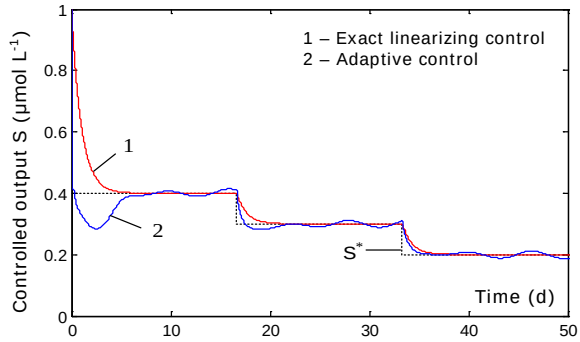

Fig. 5. Time evolution of output S – case 1

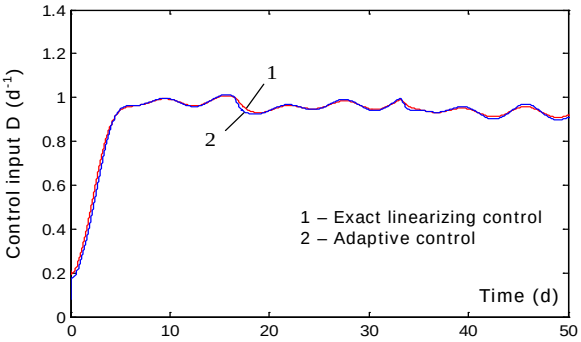

Fig. 6. Profile of control input D – case 1

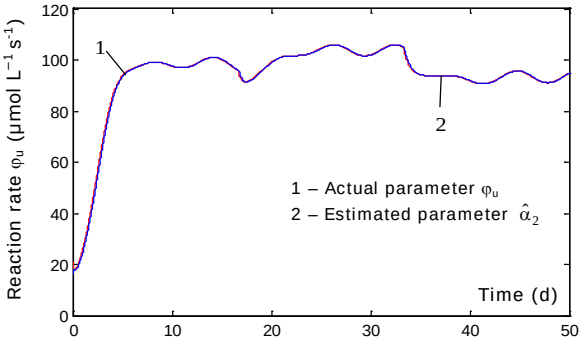

Fig. 7. Profile of estimate of unknown parameter φ_u – case 1

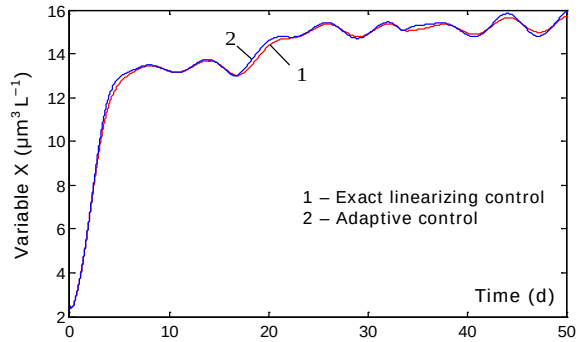

Fig. 8. Profile of variable X – case 1

The gain of control laws (7) and (4) is $\lambda = 20$, and the tuning parameters of adaptive controller have been set to: $\omega = -25$, $\gamma = 250$.

The time evolution of the estimates of unmeasured variable φ_u provided by the OBE (6) is presented in Fig. 7. From this figure it can be noticed that the parameter estimator provide a very good result.

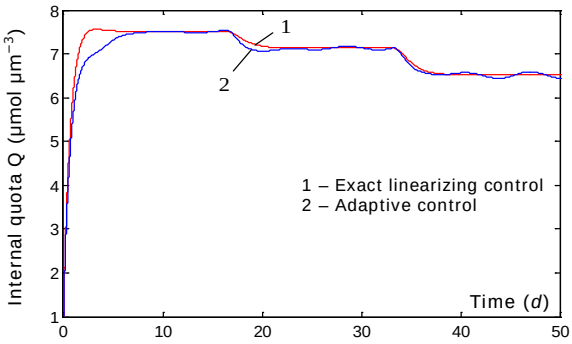

Fig. 9. Internal quota Q – case 2

In Figs. 8 and 9 are presented the time evolution of the biomass X and of the internal quote Q. From these figure one can observe that the evolution of the two variables are closed to the evolution of these variables in the benchmark case.

Graphics in Figs. 5-9 show that the behaviour of the adaptive controlled system is correct, being very close to the behaviour of closed loop system in the ideal case (completely known process model) - benchmark case - when the exact linearizing controller (4) is used. The adaptive controller is able to maintain the controlled output S very close to its desired value, in spite of the unknowing of the substrate uptake rate and of the high variation of S_{in} and of the light I.

Case 2: Predictive control. Now the performance of closed-loop system using the structure of predictive control law presented in Section 3.3 will be analysed, under the same realistic conditions about the process as in the case of the adaptive control system. The behaviour of closed-loop system using NMPC algorithm presented in Section 3.3 by comparison to the linearizing law (4) is presented in Figs. 10-14.

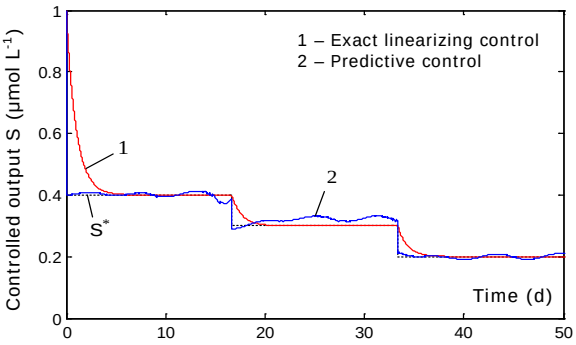

Fig. 10. Time evolution of output S – case 2

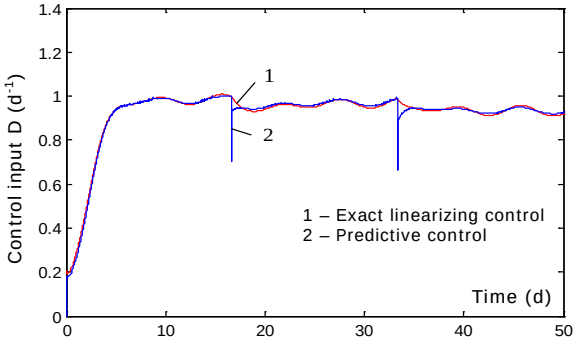

Fig. 11. Profile of control input D – case 2

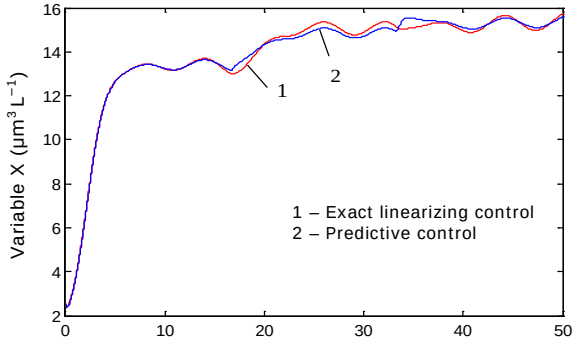

Fig. 12. Time variation of biomass X – case 2

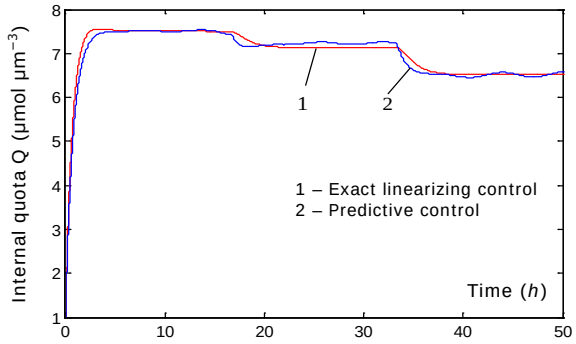

Fig. 13. Internal quota Q – case 2

The parameters used for predictive control algorithm are defined, as follows: N = 6, N_u = 6, $\Psi = I_6$, $\Omega = 0.01 \cdot I_6$, where I_6 is the 6-dimensional unity matrix, and the time sampling is T = 10 min.

Fig. 10 shows the time evolution of controlled output S, and Fig. 11 depicts the control input D. From Fig. 10 and Fig. 11 it can be observed that the substrate concentration S tracks the reference profile S^*, and the control input D is maintained in the physical limits.

The time evolution of the biomass X and of the internal quote Q are presented in Figs. 12 and 13. From these figure one can observe that the evolution of the two variables are closed to the evolution of these variables in the benchmark case.

From graphics in Figs. 10-13 it can be seen that the behaviour of overall system with the predictive algorithm is correct, being very close to the behaviour of closed loop system with adaptive controller (7) presented in *Case 1* as well as to the behaviour of closed loop system in the ideal case (exact linearizing controller).

The predictive controller present good performance to lead the system to new set-points despite of strongly time-varying of the influent substrate concentration and of the light intensity.

5. CONCLUSIONS

In this paper an adaptive and a predictive control structures for a continuous microalgae process were designed and analysed. The two control strategies are developed under the realistic assumptions that the reaction rates are strongly nonlinear, time varying and incompletely known, and the influent substrate concentration as well as the light intensity is strongly time-varying. The effectiveness of the proposed control laws was validated by numerical simulations.

The adaptive control structure was achieved by combining a linearizing control law with a parameter estimator which plays the role of a software sensor for on-line estimation of uncertain or unknown bioprocess kinetic rates.

The predictive control structure is based on Nonlinear Model Predictive Control (NMPC) strategy. The advantage of NMPC law is that it allows the current control input to be optimized, while taking into account the future system behaviour. This is achieved by optimizing the control profile over a finite time horizon, but applying only the current control input.

The two proposed control strategies were tested in realistic simulation scenarios. Taking into account all the uncertainties and disturbances acting on the process, the conclusion is that the adaptive and predictive controllers can constitute a good option for the control of such class of bioprocesses.

ACKNOWLEDGMENT

This work was supported by UEFISCDI, project BIOCON no. PN-II-PT-PCCA-2013-4-0070, 269/2014.

REFERENCES

Angulo, F. Olivar G., and A. Rincon A. (2007). Control of an anaerobic upflow fixed bed bioreactor. In *Proc. 15th Mediterranean Conf. on Contr. & Autom.*, July 27-29, 2007, Athens, Greece, Paper T04-005.

Bastin G. and Dochain D. (1990). *On-line estimation and adaptive control of bioreactors.* New York, NY: Elsevier.

Batstone D.J. et al. (2002). The IWA Anaerobic Digestion Model No 1 (ADM1). *Water Sci. & Technology*, vol. 45, no. 10, pp. 65-73.

Benattia S.E., Tebbani S., Dumur D., and Selisteanu D. (2014a). Robust nonlinear model predictive controller based on sensitivity analysis - Application to a continuous photo-bioreactor. *2014 IEEE Conf. on Control Applications (CCA), Part of 2014 IEEE Multi-conference on Systems & Contr.*, Oct. 8-10, 2014, Antibes, France, pp. 1705-1710.

Benattia S.E., Tebbani S., and Dumur D. (2014b). Nonlinear model predictive control for regulation of microalgae culture in a continuous photobioreactor. *Proc. of the 22nd MED Conference*, pp. 469–474.

Bernard O. (Responsible) (2004). *Design of models for abnormal working conditions and destabilisation risk analysis*. Report Number: D3.1b, TELEMAC IST 2000-28156, 81 pages.

Bernard O. (2011). Hurdles and challenges for modelling and control of microalgae for CO2 mitigation and biofuel production. *J. Process Control*, vol. 21, pp. 1378–1389.

Camacho E.F. and Bordons C. (2004). *Model Predictive Control*. London: Springer, 2004.

Dochain D. and Vanrolleghem P. (2001). Dynamical Modelling and Estimation in Wastewater Treatment Processes. *IWA Publ.*, 2001.

Dochain D. Ed. (2008). *Automatic Control of Bioprocesses*. UK: ISTE / John Wiley & Sons.

Eaton J.W. and Rawlings J.R. (1990). Feedback control of nonlinear processes using online optimization techniques. *Computers and Chemical Eng.*, vol. 14, pp. 469-479.

Hayakawa T., Haddad W.M., and Hovakimyan N. (2008). Neural network adaptive control for a class of nonlinear uncertain dynamical systems with asymptotic stability guarantees. *IEEE Trans. N. Networks*, vol. 19, pp. 80-89.

Ifrim G.A. et al. (2013). Multivariable feedback linearizing control of Chlamydomonas reinhardtii photoautotrophic growth process in a torus photobioreactor. *Chemical Eng. Journal*, vol. 218, pp. 191–203.

Kouvaritakis B. and Cannon M., Eds. (2001). *Nonlinear Predictive Control: Theory and Practice*. London, U.K.: The Institution of Electrical.

Logist F., Houska B., Diehl M., and Van Impe J.F. (2011). Robust multi-objective optimal control of uncertain (bio)chemical processes. *Chem. Eng. Sci*., vol. 66, no. 20, pp. 4670–4682.

Mairet F., Bernard O., Ras M., Lardon L., and Steyeret J.P. (2011). Modeling anaerobic digestion of microalgae using ADM1. *Bioresource Technology*, vol. 102, pp. 6823–6829.

Mayne D.Q., Rawlings J.B., Rao C.V., Scokaert P.O. (2000). Constrained model predictive control: stability and optimality. *Automatica*, vol. 36, pp. 789-814.

Neria-González I., Dominguez-Bocanegra A.R., Torres J., Maya-Yescas R., and Aguilar-Lópeza R. (2009). Linearizing control based on adaptive observer for anaerobic continuous sulphate reducing bioreactors with unknown kinetics. *Chem. Biochem. Eng. Q.*, vol. 23, no. 2, pp. 179–185.

Nocedal J. and Wright S.J. (1999). Numerical Optimization. Springer-Verlag Berlin, Heidelberg.

Petre E., Selişteanu D., Sendrescu D., and Ionete C. (2010). Neural networks based adaptive control for a class of nonlinear bioprocesses. *Neural Computing & Applications*, vol.19, no. 2, pp. 169-178.

Petre E., Selişteanu D., and Şendrescu D. (2013). Adaptive and robust-adaptive control strategies for anaerobic wastewater treatment bioprocesses. *Chem. Eng. J.*, vol. 217, pp. 363-378.

Petre E. and Selişteanu D. (2013). Adaptive and Robust-Adaptive Control Schemes for an Anaerobic Bioprocess with Biogas Production. *Proc. of the 17th Int. Conf. on System Theory, Control and Computing - ICSTCC 2013*, Sinaia, Romania, Oct. 11-13, 2013, pp. 404-409.

Selişteanu D., Petre E., and Răsvan V. (2007). Sliding mode and adaptive sliding mode control of a class of nonlinear bioprocesses. *Int. J. Adapt. Contr. & Signal Process*, vol. 21, no. 8-9, pp. 795-822.

Şendrescu D., Petre E., Popescu D., Roman M. (2011). Neural network model predictive control of a wastewater bioprocess, *Proc. of the 3rd Int. Conf. on Intelligent Decision Technologies (IDT'2011)*, 20-22 July 2011, Univ. Piraeus, Greece (J. Watada, G. Phillips-Wren, L.C. Jain, R.J. Howlett, Eds.), Springer-Verlag Berlin, Heidelberg, pp. 191-200.

Tebbani S., Lopes F., Filali R., Dumur D, and Pareau D. (2014). Nonlinear predictive control for maximization of CO_2 bio-fixation by microalgae in a photobioreactor. *Bioprocess Biosyst. Eng*., vol. 37, no. 12, pp. 83–97.

Tebbani S., Lopes F., and Becerra Celis G. (2015). Nonlinear control of continuous cultures of Porphyridium purpureum in a photobioreactor. *Chemical Engineering Science*, vol. 123, no. 18, pp. 207–219.

Wang, Y. and Boyd, S. (2008). Fast model predictive control using online optimization. *Proc. of the 17th World Congress of International Federation of Automatic Control (WC-IFAC2008)*, Seoul, Korea, July 6-11, 2008, pp. 6974-6979.

ANNALS OF THE UNIVERSITY OF CRAIOVA
Series: Automation, Computers, Electronics and Mechatronics, Vol. 12 (39), No. 1, 2015

Method for Anticipative Control of Bioprocess

Eugen Iancu, Emil Petre

Department of Automation and Electronics, University of Craiova,
107 Decebal Street, RO-200440 Craiova, Romania
(e-mail: eugen.iancu@automation.ucv.ro, epetre@automation.ucv.ro, http://www.ace.ucv.ro)

Abstract: The control of bioprocesses remains an inherently complex problem. The necessity to obtain good performances without installing a lot of dedicated and expensive equipment, forces the developers to use many techniques available to processing all the information that are "hidden" in the technological process. A difficulty for the design of high-performance control techniques of such living processes lies in the fact that, in many cases, the models contain kinetic parameters and/or yield coefficients that are highly uncertain and time varying. The aim of this paper is to present some possible predictive control methods, able to deal with the model uncertainties in an adaptive way, for a complex biotechnological process.

Keywords: Photoautotrophic growth bioprocess, predictive control, single exponential smoothing.

1. INTRODUCTION

Predictive control is different from a closed loop feedback control where the desired operating point is compared with an actual output point and the difference is fed back to the system. Predictive control problems are defined in their time domain, and their solution requires designing the course (future) of action so as to optimize a performance index. Using the predictive control theory we can minimize the deviations of outputs from normal levels and allows us to make future decisions.

The standard structure proposed for the anticipative control is shown in Fig. 1. The objective of this structure is to anticipate the evolution of the process and synthesize optimal controls. These commands (inputs) will be validated by testing them on the bioprocess model before the direct use to control the bioprocess.

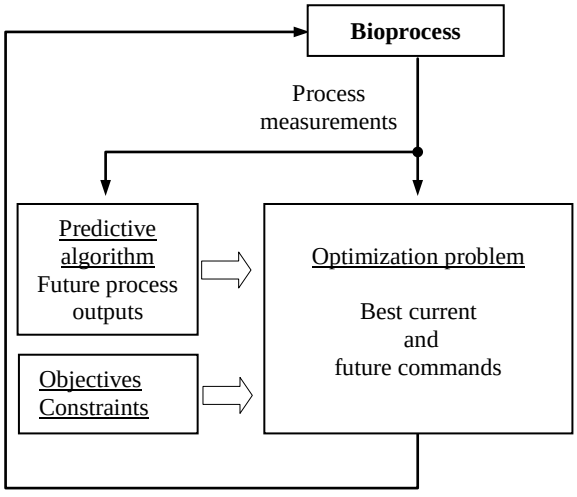

Fig. 1. The structure of the predictive control system.

2. MODEL OF THE PHOTOAUTOTROPHIC GROWTH BIOPROCESS

The wastewater biological treatment processes using anaerobic digestion is very useful since it produces valuable energy (methane) besides removing the organic pollution from the liquid influent. It is well adapted for concentrated wastes such as agricultural and food industry wastewater (Angulo et al., 2007). Nevertheless, its main drawback is the production of carbon dioxide (CO_2) and its easy destabilization, giving rise to the disappearance of the methanogenic bacteria (Angulo et al., 2007; Bastin and Dochain, 1990; Bernard, 2004).

Therefore the researchers have been searched solutions to improve the efficiency in the pollution reduction and for CO_2 mitigation (Bernard, 2011; Ifrim, 2012; Ifrim et al., 2013; Tebbani et al., 2014). A recently used solution consist in the growth of some microalgae populations that by using light as source of energy are able to assimilate inorganic forms of carbon (CO_2, HCO_3^-) and to convert them into requisite organic substances for cellular functions, generating at the same time oxygen (O_2) (Tebbani et al., 2013, 2015).

The considered bioprocess is a photoautotrophic growth of the green alga *C. reinhardtii* in a photobioreactor under light limiting conditions. Microalgae are able to absorb CO_2 as major substrate and to generate O_2 as residue from the water oxidation reaction induced by the light as source of energy (Ifrim et al., 2013; Tebbani et al., 2013). A dynamical model employed for describing all phenomena that is carried out in the photobioreactor, assuming well mixed conditions, was developed in (Ifrim, 2012). A simplified model used in (Ifrim et al., 2013), is described by the following differential equations:

$$\dot{X}^{pbr} = \langle r_X \rangle - D^{pbr} X^{pbr} \quad (1)$$

$$\dot{C}_{TIC} = -\langle r_{TIC} \rangle + N_{CO_2} + D^{pbr}\left(C_{TIC,in} - C_{TIC}\right) \quad (2)$$

$$\dot{C}_{O_2} = \langle r_{O_2} \rangle + N_{O_2} - D^{pbr} C_{O_2} \quad (3)$$

$$\dot{y}_{out}^{CO_2} = RT / PV_g^{pbr}\left(- y_{out}^{CO_2} G_{out} - V_l^{pbr} N_{CO_2} + G_{in}^{CO_2}\right) \quad (4)$$

$$\dot{y}_{out}^{O_2} = RT / PV_g^{pbr}\left(- y_{out}^{O_2} G_{out} - V_l^{pbr} N_{O_2} + G_{in}^{O_2}\right) \quad (5)$$

where X^{pbr} is the biomass concentration, C_{TIC} is total inorganic carbon concentration, C_{O_2} is dissolved oxygen concentration and $y_{out}^{CO_2}$ and $y_{out}^{O_2}$ are the molar fractions of CO_2 and O_2 in the outlet gas, respectively. $G_{in}^{CO_2}$, $G_{in}^{O_2}$ and G_{out} are the feeding gas flow rate of CO_2 and O_2, respectively and the gaseous output flux. $C_{TIC,in}$ is the influent concentration of *TIC*, and R is the universal gas constant. $V_{g/l}^{pbr}$ is the gas/liquid volume of the phobioreactor, and D^{pbr} is the dilution rate.

The average volumetric growth rate, r_X, is expressed using a Haldane model depending of the light flux into the culture medium as (Ifrim et al., 2013), (Tebbani et al., 2013):

$$\langle r_X \rangle = \left(\frac{1}{L} \cdot \mu_{\max}^{pbr} \int_0^L \frac{I(z)}{K_{ir} + I(z) + I^2(z)/K_{ii}} dz\right) X^{pbr} - \mu_s X^{pbr} \quad (6)$$

where $\mu_{\max}^{pbr}$, K_i, K_{ii} are the maximum specific growth rate, the half-saturation constant and the inhibition constant, respectively, $I(z)$ represents the local (at the depth z) value of the irradiance light flux, L the photobioreactor total depth, and μ_s the biomass death rate. For a torus photobioreactor the expression of $I(z)$ is given by (Ifrim et al., 2013):

$$I(z) = 2q_0 \frac{(1+\alpha)e^{\delta(L-z)} - (1-\alpha)e^{-\delta(L-z)}}{(1+\alpha)^2 e^{\delta L} - (1-\alpha)^2 e^{-\delta L}} \quad (7)$$

where q_0 represents the hemispherical incident light flux, is the extinction coefficient and $\alpha = \sqrt{E_a/(E_a + 2bE_s)}$ the linear scattering modulus with E_a is the mass absorption coefficient, and E_s is mass scattering coefficient (Ifrim et al., 2013).

Because the microalgal growth is synthesized exclusively from inorganic carbon, the *TIC* consumption rate is given by (Ifrim et al., 2013):

$$\langle r_{TIC} \rangle = (1/M_x) \cdot \langle r_X \rangle \quad (8)$$

where M_x is the C-mole mass of the cells. The O_2 production rate is proportional with r_X and can be expressed through the photosynthetic quotient K_P, as:

$$\langle r_{O_2} \rangle = (K_P / M_x) \cdot \langle r_X \rangle \quad (9)$$

The CO_2 and O_2 mass transfer rates to the liquid phase are expressed as follows (Tebbani et al., 2013):

$$N_{CO_2} = k_L a_{CO_2}\left(P / H_{CO_2} y_{out}^{CO_2} - C_{CO_2}\right) \quad (10)$$

$$N_{O_2} = k_L a_{O_2}\left(P / H_{O_2} y_{out}^{O_2} - C_{O_2}\right) \quad (11)$$

where $k_L a_{CO_2}$ and $k_L a_{O_2}$ are volumetric mass transfer coefficients, H_{CO_2} and H_{O_2} are the Henry constant at 25°C and P the pressure of CO_2 and O_2, respectively. Since CO_2 concentration is not measured then it is calculated from the TIC concentration and pH as follows (Tebbani et al., 2013):

$$C_{CO_2} = C_{TIC} / \left(1 + K_1 10^{pH} + K_1 K_2 10^{2\,pH}\right) \quad (12)$$

where K_1 and K_2 are the equilibriums constants.

3. EXPONENTIAL SMOOTHING METHOD

Single exponential smoothing is used for smoothing discrete time series. The efficiency of this algorithm can be attributed to its simplicity and to the capacity to adjust its responsiveness to changes in the process and its reasonable accuracy.

Let be an observed time series $X = \{x_1 \quad x_2 \quad \ldots \quad x_n\}$. Formally, the simple exponential smoothing equation takes the form (Ostertagová, 2011):

$$\tilde{x}_{i+1} = \alpha x_i + (1-\alpha)\tilde{x}_i \quad (13)$$

where x_i is the actual, known series value at moment time i, $\tilde{x}_i$ is the forecast value of the variable X at time i, $\tilde{x}_{i+1}$ is the forecast value at time i+1 and α is the smoothing constant. Smoothing constant α is a selected number between zero and one, $0 < \alpha < 1$ (Brown and Meyer, 1961). When $\alpha = 1$, the original and smoothed version of the series are identical. At the other extreme, when $\alpha = 0$, the series is smoothed flat (Ostertagová, 2011). In the literature it is demonstrate the next relation (Brown and Meyer, 1961):

$$\tilde{x}_{i+1} = \alpha x_i + \alpha(1-\alpha)x_{i-1} + \alpha(1-\alpha)^2 x_{i-2} + \ldots + \alpha(1-\alpha)^{i-1} x_1 = \alpha \sum_{k=0}^{i-1} (1-\alpha)^k x_{i-k} \quad (14)$$

In scientific papers are presented also *double exponential smoothing* and *triple exponential smoothing*.

From (13) we obtain:

$$\tilde{x}_{i+1} = \tilde{x}_i + \alpha(x_i - \tilde{x}_i) = \tilde{x}_i + \alpha\varepsilon_i \quad (15)$$

where ε_i represent the forecast error at time i. Using this

error it is possible to define the following parameters (Ostertagová, 2011):

- Mean square error - *MSE*

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\varepsilon_i^2 \quad (16)$$

- Root mean square error - *RMSE*

$$RMSE = \sqrt{MSE} \quad (17)$$

The objective is to find an appropriate smoothing constant so that *MSE* and *RMSE* to be minimum.

4. THE STRUCTURE PROPOSED FOR CONTROL OF THE PHOTOAUTOTROPHIC GROWTH BIOPROCESS

The simplified structure of the photoautotrophic growth bioprocess is shown in the Figure 2.

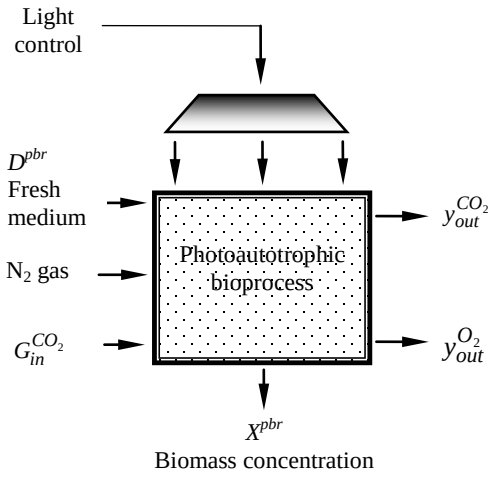

Fig. 2. The structure for the photoautotrophic bioprocess.

The bioreactor is characterised by relatively large time constants and the growth bioprocess can be characterised by the presence of a dead time. This is why it is necessary to use an anticipatory control algorithm. The proposed structure is shown in Fig. 3. The significance of the notations used in the structure shown in Figure 3 are the following:

- $G_{in}^{CO_2}$ - is the feeding gas flow rate of CO_2
- $G_{in}^{CO_2+}$ - is the predicted feeding gas flow rate of CO_2
- D^{pbr} - is the dilution rate
- D^{pbr+} - is the predicted dilution rate

The assembly contains:

- The real bioprocess (photoautotrophic)
- The mathematical model of the bioprocess
- The predictive algorithm for CO_2 gas flow rate
- A structure for decisions and finally, for synthesis and validation of command for fresh nutrient substrate (D^{pbr+} calculated).

The roll of the predictive algorithm is to anticipate the values of the $G_{in}^{CO_2}$ gas flow rate of CO_2 and to enable early synthesis of the values for D^{pbr+}.

The structure of decision must to synthesize values for the D^{pbr+} command so $y_{out}^{CO_2}$ approached to zero. This indicates that the biomass has a growth rate which should lead to the total consumption of CO_2.

For optimal control, additional it can make a correlation between $y_{out}^{O_2}$ and nutrient flow. Thus, it is possible to have one of the following situations:

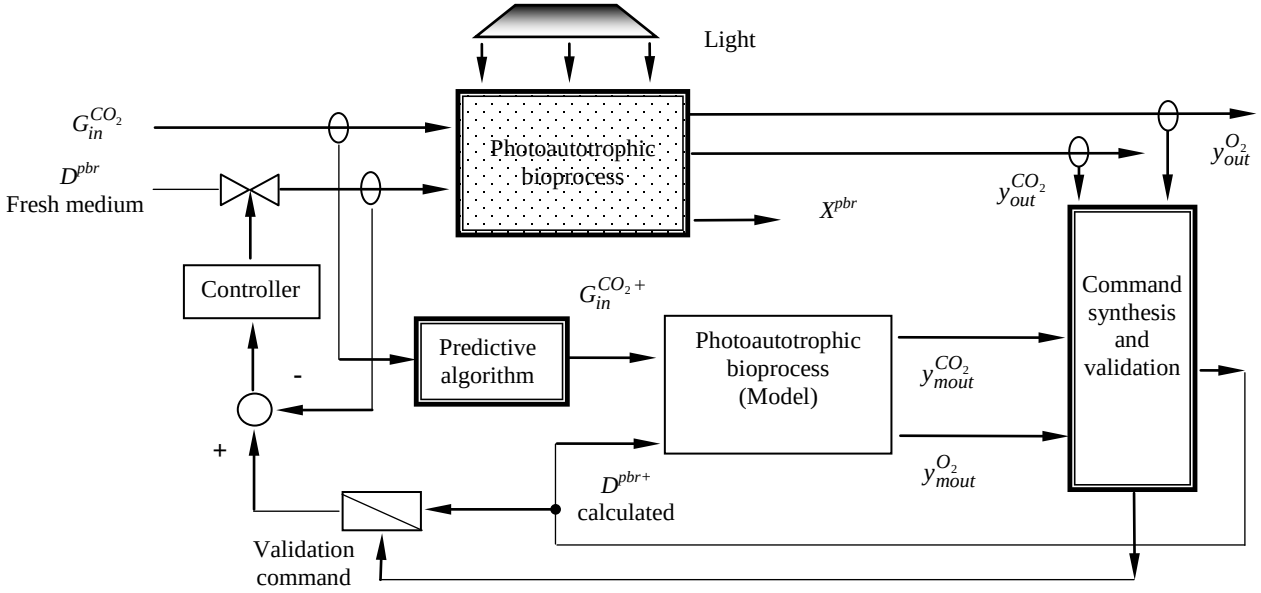

Fig. 3. The structure for control system with predictive algorithm.

1. Assume that the estimated flow of $G_{in}^{CO_2}$ it is a constant. The increase of the flow of nutrient concentrations lead to the decreased by $y_{out}^{CO_2}$ and an increase in the concentration of O_2 ($y_{out}^{O_2}$). Decides to increase on further the flow of nutrient (D^{pbr+}). As long as the $y_{out}^{O_2}$ increases, the procedure continues in the same way.
2. Assume that the estimated flow of $G_{in}^{CO_2}$ decrease. This evolution is accompanied by decrease of the microalgal growth and consequently of $y_{out}^{O_2}$. In this case we must decrease also the flow of nutrient concentrations so $y_{out}^{CO_2}$ as to be close to zero (action is helpful in terms of consumption).
3. Assume that the estimated flow of $G_{in}^{CO_2}$ increase. This evolution is accompanied by increase of the microalgal growth and consequently of $y_{out}^{O_2}$. In this case we must increase also the flow of nutrient concentrations so $y_{out}^{CO_2}$ as to be close to zero.

Recommendation: because there is a degree of uncertainty bioprocess modelling, preventive it is recommended to be excessive nutrient substrate. Also, it is worth mentioning that in the synthesis of the control law, we must consider the ensuring of closed-loop stability.

The authors applied the method of single exponential smoothing to the time series that simulates the input flow of CO_2, for α = 0.2, 0.6 and 0.9. $G_{in}^{CO_2}$ (Fig. 4).

5. CONCLUSIONS

The objective control of the presented photobioreactor is to minimise the quantity of CO_2 in the output flux G_{out} and to increase the quantity of O_2 as well as to obtain some non-pollutant components under conditions when the feeding gas flow rate of CO_2 as well as the light flux are time varying and the dilution rate D^{pbr}.

The control system needs to adopt the correct attitude, to know the good commands, so he must choose and validate the correct decision. This is the role of the anticipative simulation of dynamic system in these situations.

ACKNOWLEDGMENT

This work was supported by UEFISCDI, project BIOCON no. PN-II-PT-PCCA-2013-4-0070, 269/2014.

REFERENCES

Angulo, F., Olivar, G. and Rincon A. (2007). Control of an anaerobic upflow fixed bed bioreactor, in *Proc. 15th Mediterranean Conf. on Contr. & Autom.*, July 27-29, Athens, Greece, Paper T04-005.

Bastin, G. and D. Dochain D. (1990). *On-line estimation and adaptive control of bioreactors*, New York, NY: Elsevier.

Bernard, O., (Responsible), (2004). *Design of models for abnormal working conditions and destabilisation risk analysis*. Report Number: D3.1b, TELEMAC IST 2000-28156, 81 pages.

Bernard, O. (2011). Hurdles and challenges for modelling and control of microalgae for CO_2 mitigation and biofuel production, *J. Process Control*, vol. 21, pp. 1378–1389.

Brown, R.G. and Meyer, R.F. (1961). The fundamental theory of exponential smoothing. *Operations Research*, vol. 9, pp. 673-685.

Ifrim, G.A. (2012). Control of Two Biological Processes of Environmental Interest (Biological Wastewater Treatment and Microalgae Production in Photobioreactor), PhD thesis, University of Nantes, France or University "Dunarea de Jos", Galati, Romania.

Ifrim, G.A. et al. (2013). Multivariable feedback linearizing control of Chlamydomonas reinhardtii photoautotrophic growth process in a torus photobioreactor. *Chemical Eng. Journal*, vol. 218, pp. 191–203.

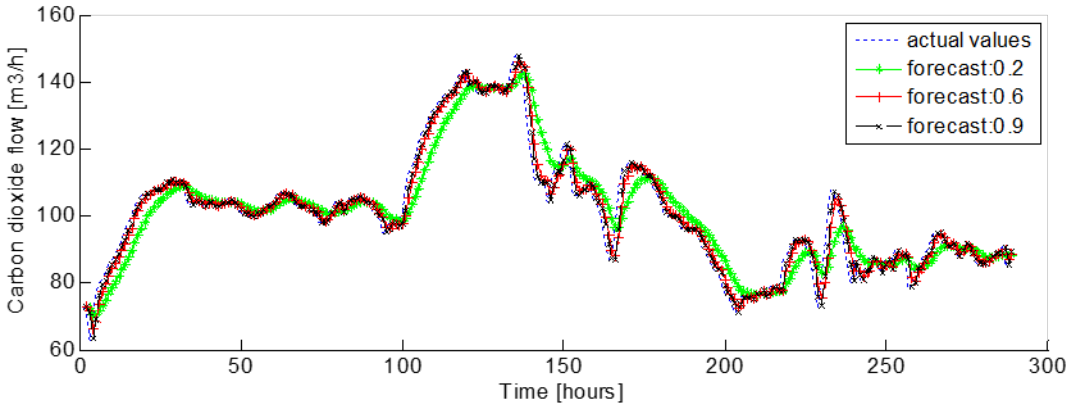

Fig. 4. The values of CO_2 flow with predictive algorithm (simulation).

Ostertagová, E. and Oskar Ostertag, O. (2011). The simple exponential smoothing model, The 4th International Conference on Modelling of Mechanical and Mechatronic Systems, Technical University of Košice, Slovak Republic, *Proceedings of conference*, pp. 380-384.

Tebbani, S., F. Lopes, R. Filali, D. Dumur, and Pareau D. (2014). Nonlinear predictive control for maximization of CO_2 bio-fixation by microalgae in a photobioreactor, *Bioprocess Biosyst. Eng.*, vol. 37, no. 12, pp. 83–97.

Tebbani, S., F. Lopes, and BecerraCelis G. (2015). Nonlinear control of continuous cultures of Porphyridium purpureum in a photobioreactor, *Chemical Engineering Science*, vol. 123, no. 18, pp. 207–219.

Tebbani, S., Titica, M., Caraman, S., and Boillereaux, L. (2013). Estimation of Chlamydomonas reinhardtii Growth in a Torus Photobioreactor, *Preprints of the 12th IFAC Symposium on Computer Applications in Biotechnology*, 16-18 Dec., Mumbai, India, pp. 155–160.

ANNALS OF THE UNIVERSITY OF CRAIOVA
Series: Automation, Computers, Electronics and Mechatronics, Vol. 12 (39), No. 1, 2015

Improving of the Backtracking Algorithm using different strategy for solving the 2-d problems

Bogdan Popa, Dan Popescu

Department of Automation and Electronics, University of Craiova,
107 Decebal Street, RO-200440 Craiova, Romania
(e-mail: bogpo89@yahoo.com; dpopescu@automation.ucv.ro)

Abstract: Nowadays, many algorithms in the field of artificial intelligence are based on the backtracking principles. These algorithms require highly efficient systems due to the high cost of execution time of solving backtracking, significant adjustments are needed to optimize these complex methods. Whether parallel programming or differentiated approach to the problem can bring better results of such as algorithms that responds to all possible solutions to a scenario. This article illustrates a method to improve backtracking algorithm, depending on the problem solved by using new systems for software development. This article can offer a new perspective over the mapping systems for different fields where there we need to find all the solutions and also to give different contexts to the searching for the solutions. The results of this study consisted in methods for give a better time execution for finding the solutions for the proposed problems and an analysis for every strategy chosen.

Keywords: Backtracking algorithm, Artificial intelligence, Improving algorithms, Strategy for 2-d problems.

1. INTRODUCTION

Backtracking is a general algorithm for finding all solutions to some computational problems, notably constraint satisfaction problems that incrementally builds candidates to the solutions, and abandons each partial candidate c ("backtracks") as soon as it determines that c cannot possibly be completed to a valid solution (Knuth, 1968; Gurari, 1999). Another application is the Backtracking Search Algorithm (BSA) that is widely used to solve parameter estimation, satellite formation flying control, optimal design of antenna arrays (Witten et al., 2005). For example, the main problem that this algorithm solves is the 8 queens problems on a chess table. Also can generate some similar problems to place various pieces on the chessboard because the Backtracking system offers solution for every permutation scheme. This proposed algorithm is often much faster than brute force enumeration for all complete solutions, since it can eliminate a number of phases with a single test.

Backtracking is today used for solving constraint problems, like verbal arithmetic, puzzles types problems; it is also the most convenient technique for parsing and for combinatorial optimization problems. It was considered the basis of logical programming languages such as Icon, Planner and Prolog. Today there are discussed two hybrid backtracking algorithms, Backtracking with Backjumping (BMJ) and Backmarking with Conflict-Directed Backjumping (BM-CBJ), so that they always perform fewer consistency checks than the original algorithms. The idea is to make fewer steps and to obtain all the solutions in minimal time. There is no simple answer to the question which backtracking algorithm is the best one. First, the performance of backtracking algorithms depends heavily on the problem being solved. Also, the backtracking method is part of the artificial intelligence domain and like in this article, it can offer requested solutions for 2-D systems. Run time is not a very reliable measure because, nowadays it depends on hardware and implementation, on method used and is directed influenced by another factors, but the strategy and the algorithm can give different results depends on the scope. Backtracking is one of the simplest applications for the technique in problem solving named “divide and conquer”. The prototypical backtracking problem is the classical N Queens Problem, first proposed by German chess enthusiast Max Bezzel in 1848 (under his pseudonym “Schachfreund”) for the standard 8×8 board and by François-Joseph Eustache Lionnet in 1869 for the more general N×N board (Lionnet E., 1869).

2. BACKGROUND

A constraint satisfaction problem (CSP) consists of a set of variables, $X = \{x_1,...,x_n\}$; a set of values, $D = \{a_1,...,a_d\}$, where each variable $x_i \in X$ has an associated finite domain $dom(x_i) \subseteq D$ of possible values; and a collection of constraints. A good example for testing and develop the Backtracking algorithm is the N-Queens problem that can represent a 2-D case for constraints, according to the chess rules for every piece. This example can be applied to another type of chess piece. Let’s consider the 8-queen problem, which is computationally very expensive since the total number of possible arrangements queen is 64!/(56!x8!) ~ 4.4 x 109 and the total number of possible solutions are 92 (Kesri et al.,

2012). But there are effectively only 12 unique solutions. There are some solutions that are to be the same and can be obtained one from the other by taking rotations or symmetry. The Queens' problem belongs to the class of problems called "Embarrassingly parallel": the solution to one subproblem (here represented by the queen's position in the first row) is completely independent of another subproblem (a different position in the first row) (Rolfe, 2004).

Table 1. The number of total and unique solutions for the N-Queens problem

Number:	Unique Solution	Distinct Solution
1	1	1
2	0	0
3	0	0
4	1	2
5	2	10
6	1	4
7	6	40
8	12	92
9	46	352
10	92	724
11	341	2680
12	1787	14200
13	9233	73712
14	45752	365596

In Table 1 it is presented the number of unique solution and the number of distinct solution for every example, taking into account the Backtracking algorithm for N-Qeeens example.

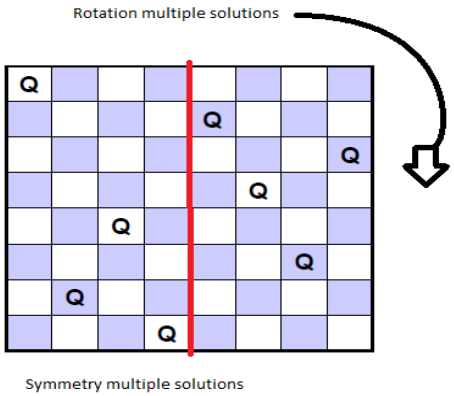

Fig. 1. Rotation and Symmetry multiple system for generating multiple solutions

The process of Backtracking algorithm is to extend partial solutions at every step. At every step of backtracking search, there are some partial solutions which the algorithm expects to complete to the full solution. The basic algorithm is the next one:

```
PROCEDURE BT(k)
FOR Z[k] = 1 TO m[k] DO
  BEGIN
  Consistent = True;
  FOR p = 1 TO k-1 WHILE consistent DO
                          consistent = check(p, z[p],k, z[k]);
  IF consistent THEN IF k = n THEN output(z) ELSE BT(k+1)
  END;
END;
```

For the sequential Backtracking algorithm the time is time expensive, depends of the number of lines for the chess example.

Board Size	Number of solutions	Execution Time to find all solutions (ms)
1	1	0
2	0	0
3	0	0
4	2	4
5	10	11
6	4	19
7	40	29
8	92	44
9	352	93
10	724	210
11	2680	421
12	14200	2376
13	73712	13663
14	365596	85992
15	2279184	523597

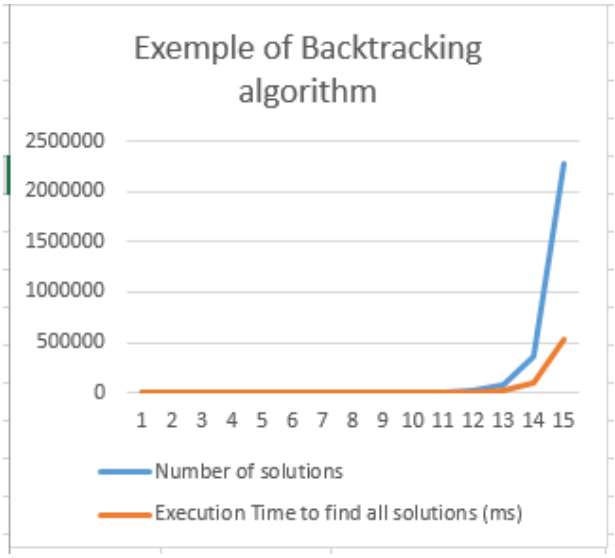

Fig. 2. Testing the Backtracking algorithm for N-Queens example

It is obvious that the number of solutions goes up roughly exponentially with the size of the board. Starting to this example in this article we will propose new models for giving better strategy to make the improvements in the N-Queens problem using Backtracking. In the firs time, the Backtracking method is better than Brute-force methode which is redundant in this cases, but also this is time expensible. Over the 13 or moore queens the execution time starts the innacesible way for instant systems. In the Fig. 1 there are some strategy for define the differences

between the Unique and Distinct solution taking into account the 90 degree, 180 or 270 degree rotation from a unique solution, also there are the Vertical mirror, Anti-diagonal mirror, Horizontal mirror and the Diagonal mirror methods to convert a unique solution to multiple solutions.

3. PROPOSED STRUCTURE FOR BETTER RESULTS

Starting from the execution time high cost, the article proposes new perspective over the 2-d problems for using backtracking algorithm.The improvement is related to three directions:

1) Rotation and Symmetry applied to a unique solution found; in this case we can expand the results from one solution to 8 distinct solutions.
2) Utility of the Backjumping strategy, for performing fewer steps in the searching for the unique solution.
3) Using better computing systems over the parallel calculations if there are possible.

Backjumping (BJ) (Gaschnig, 1978) is similar to BT, except that it behaves more efficiently when no consistent instantiation can be found for the current variable X[i]. Instead of chronologically backtracking to the preceding variable, BJ backjumps to the deepest past variable X[h] that was checked against the current variable. In the proposed scenario the solution will backjump 2 stages before, from test the consistency is loose and about 10 percents of solution are not found.

The proposed scheme using these 3 betters performing strategy convert to the next type of structure:

```
PROCEDURE BT(k)
FOR Z[k] = 1 TO m[k] DO
  BEGIN
  Consistent = True;
  FOR p = 1 TO k-1 WHILE consistent DO
                        consistent = check(p, z[p],k, z[k]);
  IF consistent THEN IF k = n THEN output(z) (inset parallel scenario)  ELSE BT(k+1)
  END;
END;
```

The new parallel scenario compute faster the maximum 8 new solution from symmetry and starts new threads for every of these solutions and this process is multiplied with maximum 8 solutions and threads at every new solution found. It must be said that in every new solutions will be applied Backjumping with 2 stage steps. Starting of these scenario proposed, it is obvious that we will find a lot of identical solutions because every independent thread it can find a solution previous proposed by another threads route. A solution for this overlay of the identical solution consist in a saving system of solution passed and the check system for validity at every step, but it will be a lot of time and memory used for this strategy.

The problem consists in the order perturbation calculation that offers different solution. From another point of view it can be said that this proposed scenario offer a faster set of solutions but also a lot of redundant calculus. The results of these N-Queens problem is presented in the next chart and also the better performance for simple backjumping solutions implemented (for not all the solution offered), and for the hybrid Backtracking using the previous 3 concepts.

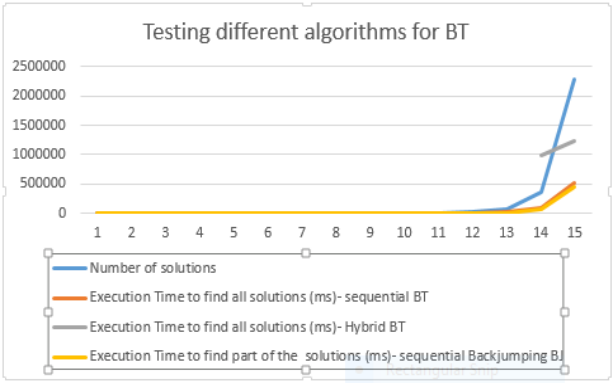

Fig. 3. Testing the Backtracking algorithm for N-Queens example with multiple implementation strategy

From the predict stage, it is obvious that the hybrid proposed solution have failed for the sped-up process. For better result it may be inserted into the algorithm a stop point under the searching solution stage under the identical search points. A good implementing operation consist in a flags method for preventing the same calculation, but also this method can be difficult to implement or to test in reasonable time.

In another study offered by Rolfe (2004), there are presented some good improvements over the static BT algorithm taking into account the validity Check system proposed by Niklaus Wirth, in his Algorithms and Data Structures (Wirth (1986), showed that it can be performed the check in constant time, provided that the use of some additional arrays to hold information.

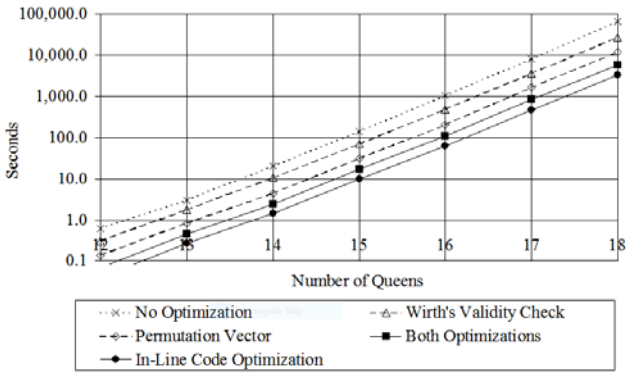

Fig. 4. Better optimization of the Backtracking algorithm for N-Queens example with multiple implementation strategy (Rolfe, 2004)

From the previous figure, and the Timothy Rolfe proposed optimizations, results that are some techniques such as permutation vector, Wirth Validity Check that offer good perspective over the BT algorithm, with a factor of speed-up for the algorithm. It is done it proportional, with a constant factor of optimization at every test case, and gives the best structure with a hybrid algorithm with the both implementations in the same time. Wirth's algorithm dramatically speeds up the processing, requiring only about half the time as compared with the code using Horowitz and Sahni's order-N validity check (Horowitz and Sahni, 1978) .

As implemented by Horowitz and Sahni (1978), the process of checking for the validity of a position within a given row explicitly checks each previous row. Also the Horowitz and Sahni optimization can be introduced in proposed strategy because it speed-up the results. The representation of a board configuration can be given by a vector Map in which each cell gives the column position of the queen on that row. The check is thus an O(N) operation and is itself quite short. Row is gives the row being checked (Wirth, 1986), Idx, the rows above:

```
for (Idx = 0; Idx < Row; Idx++)
  if ( Map[Idx] == Map[Row] ||
      abs(Map[Row]-Map[Idx]) == (Row-Idx) )
        return 0;
return 1;
```

Starting to this type of optimization, it can be said that adding this strategy it should be obtained better result with ¼ less time and also offer good example for parallel possible implementation.

Another parallel solution to this algorithm can be applied with simple positioning with the generation of all solutions with a queen placed in the first column of the first row, does not require any information from the solutions with the queen in another column of the first row and this can be an independent case for another placement for the queen on the first row and starting from this example we can open new calculation threads for solutions. It means that we establish a configuration from different starting positions and calculate the solutions in parallel, but just for a half of the table because the another half can be processed with the rotation and symmetry computing for the solutions. This proposed section should offer theoretical from the stars a half of time for a half of table position checking and these half can be processed in parallel for the first row, for example for 8x8 table we should give 4 different threads that are working in parallel.

The critical part executed on row/2 parallel time starting with the first step by the positioning of the queen on the first row.

```
void queen(int row, int n)
{
     int column;
     for(column=1;column<=n;++column)
     {
     if(place(row, column))
     {
           map[row]=column;
           if(row==n)
         { print(n);     ---->save the configuration for
multiplication process by symmetry     }
     else //try queen with next position
     queen(row+1,n);
      }
 }
}
```

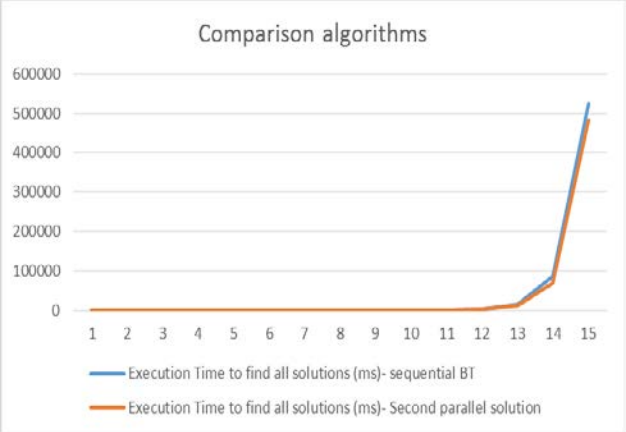

Fig. 5. The comparison over the sequential and the last parallel proposed solution

In this case the result are expected, because the parallel method should work better but there are a lot of factors that influence the results such as the type of processor, also the memory used for saving and convert the solutions, also the technology used and a lot of different factors. But, not at theoretical expectances, but the parallel hybrid solution is working better.

4. FUTURE WORK

For the future an interesting idea should be to reuse the first algorithm, to use Backjumping in parallel and also Backmarking for better results. Also a new possibility will be to use another technology such as MPI (Message Parsing Interface), for the implementations because in this medium, with multiple processors the results will look spectacular. There are a lot of improvement that can be added to the combinatorial algorithms using parallel programming, symmetry solutions finder and more distinct speed-up systems. Also a new field of study can be considered applications of Grover's quantum algorithm that solves an unstructured search problem in a time of order $\sqrt{N}$, where N is the dimension of the search space, which corresponds to a quadratic speed-up over a classical search (Grover, 1996). This algorithm is proved

to be optimal in the case of unstructured search problems (Zalka, 1999).

5. CONCLUSIONS

This article proposes a parallel implementations form over the Backtracking algorithm applied with the N-Queens problem. For example, Nadel (1990) uses the N-queens problem to show that there are always alternative CSP models of a problem and that, given the naive chronological backtracking algorithm; these models can result in different problem solving performances. These type of solution are today used in different fields like map search strategy, password finder (better than brute force), with extensive applications in Artificial Intelligence, Electronic Design Automation, and many other fields of Computer Science and Engineering, in techniques for fraud detection in financial applications. Also a new hypothesis about the this type of problems, for map finder solutions is presented taking into account the importance of the processing time that is expensive today. In complex applications that contains Backtracking strategy, for a big numbers of data and also a big number of solutions, there are some critical time consumption points that must be converted to better strategy and to improve the classical algorithms how much it can be done. Another interesting proposed extension of this application can be to create a 3 D dimension algorithm for CSP proposed problems and to use some new technology on this example with the improvements over the Backtracking algorithm for this new challenge. With applications in artificial intelligence domain, as well as the area for future backtracking method implementations, remains one of the techniques use today in various fields, but also with a great opportunity to expand into new areas. Examples presented, also provide comprehensive new proposals on algorithm improvement method for cases with large number of iterations.

REFERENCES

Knuth, D. (1968). *The Art of Computer Programming*, Addison-Wesley.

Gurari, E. (1999). *Backtracking algorithms*, CIS 680: Data structures: Chapter 19: Backtracking Algorithms.

Witten, H.I., Frank, E., and Hall, A.M. (2005). *Data Mining: practical Machine learning Tool an techniques*. Morgan Kaufmann Publisher, Burlinghton.

Lionnet, E.(1869), *Nouvelles annales de mathematiques*, serie 2, vol. 8, p. 560.

Kesri, Vi., Kesri, Va., and P. Ku. Pattnaik, Ku.P. (2012). An unique solution for N queen problem. *International Journal of Computer Applications*. Vol. 43, Issue 12, pp. 1-6.

Rolfe, T. (2004). The Optimal Queens. *Dr. Dobb's Journal*, vol. 30, no. 5, May 2004, pp. 32-37.

Gaschnig, J (1978). Experimental case studies of backtrack vs. waltz-type vs. new algorithms for satisficing assignment problems. In *Proc. of the 2nd Biennial Conference of the Canadian Society for Computational Studies of Intelligence*, pp. 268–277.

Wirth, N. (1986). *Algorithms and Data Structures*, Prentice Hall.

Horowitz, E. and Sahni S. (1978). Fundamentals of Computer Algorithms. *Computer Science Press*, pp. 324-39.

Grover, L.K. (1996). A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual Symposium on the Theory of Computing* (ACM Press, New York), pp. 212–219.

Zalka, C. (1999). Grover's quantum searching algorithm is optimal. *Phys. Rev*. A60, 2746-2751, 13 pag.

Nadel, B.A. (1990). Representation selection for constraint satisfaction: A case study using n-queens. *IEEE Expert*, vol. 5, pp. 16-23.

ANNALS OF THE UNIVERSITY OF CRAIOVA
Series: Automation, Computers, Electronics and Mechatronics, Vol. 12 (39), No. 1, 2015

An implementation in BlueJ used in teaching object-oriented programming

Adrian Runceanu

Department of Automation, Energy, Environment and Sustainable Development
Constantin Brancusi University of Targu-Jiu
(e-mail: adrian_r@utgjiu.ro)

Abstract: In this paper we present a visual programming environment oriented, BlueJ, in which we can built it using applications developed with classes. The approach in this paper proposes to use interactive teaching in BlueJ. In this way the student can better understand concepts related to object-oriented programming (OOP) and especially may make changes, improvements at the implemented applications. The paper is implementing the Tower of Hanoi problem, practical application of the method of Divide and Conquer programming.

Keywords: Object-oriented programming, Java, BlueJ, Divide et Impera, Recursion

1. INTRODUCTION

Most of the students who are studying at the Automation and Applied Informatics (AIA) degree program, believe that the course of Object Oriented Programming (OOP) is one of the most difficult teaching activities. The teachers who teach this course tries various methods to improve outcomes in terms of learning these methods of design applications. These techniques include:

- Choosing the first programming language
- Choosing a book / manual course
- Choosing a development environment applications
- Different teaching methods
- Different ways of allocating / evaluation topics / projects for students

As I studied I have not found a recipe for success that can be applied in teaching OOP so that more students to improve their knowledge in this field. Choosing a programming language through which they can teach the notions of classes and objects, such as C ++, it resulted in small measure getting better results by the student, likewise, the choice of a manual / course very well developed. Some teachers, including myself, have tried to lower the standards and reduce the amount of information and focus on certain chapters which they considered most important (Baldwin and Kuljis, 2000).

Among the students that study at AIA we have found a wide variety of skills, speed and attitude study vis-à-vis this course. Since the attention to the difficulties that a student face cannot be addressed easily in a group of students, the teachers must find an alternative way for a helpful learning. Therefore, we try to use a variety ways of learning.

The Department of Automation, Energy, Environment and Sustainable Development (AEMSD) from Constantin Brancusi University offers a licence study program called Automation and Applied Informatics (AIA). In this program of study the students must learn in their first year Computers Programming and Algorithms Design, and in the second year about OOP. Typically the students do not get good grades at this three subjects compared to other subjects in the two study years.

Every time we start a new school year, we try to see what didn't work last year, so to make the necessary changes in the course structure, in the assessment, etc. We strive to make improvements in teaching methods that we use in choosing books / courses better than handouts and choose programming environments suitable on the one hand to our students and on the other hand depending on the updates in programming field.

We evaluated several textbooks and we found that most of them follow a standard procedural approach of Java presentation. The programming concepts are usually filled with primitive data types, control statements, repetitive structure presented before proposing notions of classes, objects and methods. A typical first program is the program "Hello World", which is not really suitable for introduction in object oriented programming (OOP) and this because the input-output (I/O) operations in Java require a good knowledge of language. Many textbooks present special development environments and with special customized classes of I/O. Thus the students become disoriented using these classes for the I/O operations. From experience we have noticed that the students differentiate very hard the basic concepts of OOP, namely: class methods and instance methods, or variables and class instance variables.

The main reason that we want to modify each year how to teach OOP is that most students only use Java as a procedural language, and hence cannot understand object orientation programming (OOP). They are not used to solve problems by designing writing classes, but by editing programs monolithic in the main class. So they cannot understand two basic concepts in OOP namely class and class instance. And here other concepts like inheritance and polymorphism that are difficult to

understand, or even a mystery to them. We want every time that students truly understand the concept of object, but it is difficult to use Java syntax because there are details that must be mastered better so to understand this notion (Barnes and Kölling, 2003).

We have studied the manual-book proposed by Barnes and Kölling (2003) on the development environment BlueJ (http://www.bluej.org) and we found that the two authors were able to focus on eliminating the weak points other hand and to provide good general concepts of OOP, into an integrated visual user-friendly development environment. BlueJ is an interactive development environment designed specifically as a medium of learning OOP at Monash University in Australia. BlueJ main features are:

- Provides a view of the class structure. This displays a UML diagram type representing grades and relationships from the created project (see Fig. 1).

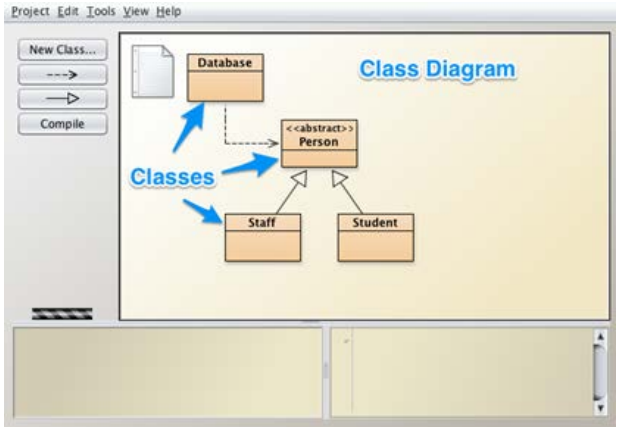

Fig. 1. Class diagram in BlueJ environment.

- The users can directly create objects of any class Fig. 2), and can then interact with the methods of that class (see Fig. 3). They can try a method immediately after being written without having to write some test sequences. This feature is invaluable in understanding OOP concepts and details of language.

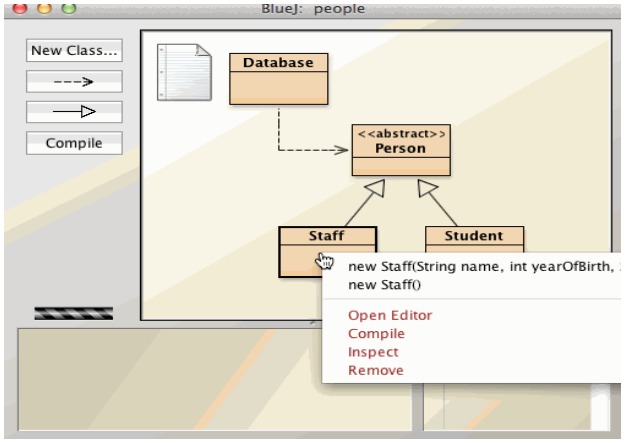

Fig. 2. Creating an object in BlueJ.

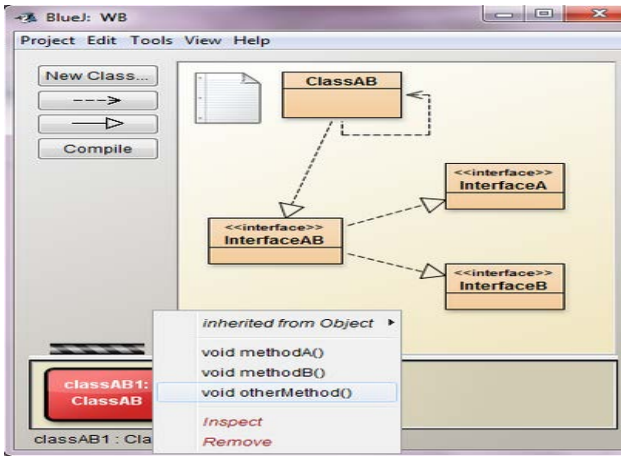

Fig. 3. Calling a method on an object in BlueJ.

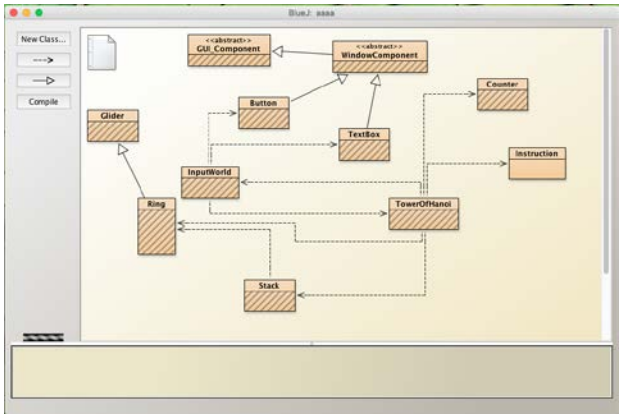

Fig. 4. UML diagram of our project in BlueJ.

1.1. Advantages of the BlueJ Environment

BlueJ is designed to facilitate learning of object-oriented concepts. Its visual environment makes it possible for novice students to interact with existing classes without writing any code at all, for the first few weeks of their programming course, until they understand the basic concepts of object-oriented programming. However, it is necessary for them to learn how to write Java code by the end of the semester. BlueJ's editor (see Figure 2) helps them to do this, and to debug their programs. In BlueJ, objects can be instantiated from classes and inspected to ascertain their state (see Figure 4), and requested to execute their public methods. Returned values from methods are shown in a window, and output to the screen in another window. Students are prompted for values of method arguments when appropriate. Classes are depicted as rectangles in the main window, and objects are shown as ovals in the object window at the bottom of the screen (see Fig. 1).

A class has a dark border around it if it has compiled successfully, and is striped otherwise (see Figure 3). The student clicks on a class to select it, then a right mouse click brings up a menu of all the public methods of that class. The student can select a constructor to create an object, and is given the opportunity to name the new

object. If it is a constructor that requires arguments for the values of the attributes of the new object, the student is prompted to key in those values. The new object is then shown in the bottom window, labelled with its name. It is not necessary for students to write a main method for a program, as they can ask an object or a class directly to execute any of its public methods.

However, it is possible to include a main method in a program, if that is desired. When classes are brought together from various directories into the BlueJ environment to construct a program, BlueJ automatically creates a package, and students rarely have to use the import statement in their programs. The BlueJ environment makes testing a class easy, as students can invoke each method in the class directly and see the output and returned value. They can create a method to test all the other methods systematically, and invoke it directly. They can also include such a method in a test class whose purpose is to test another class thoroughly. (Hagan and Markham, 2000a; Hagan and Markham, 2000b)

2. MOTIVATION

The Object-Oriented Programming (OOP) course covers fundamental concepts of object-oriented programming. The topics include data abstraction, classes and objects, state and behaviour, methods, inheritance and subtyping, polymorphism and software reuse, overloading and overriding, dynamically-bound method calls, and data encapsulation. These concepts can be rather difficult to grasp for beginning programmers. Furthermore, students often have difficulty understanding the reason for and advantages of using the object-oriented approach to software development. I use some application in Java with visual and animation (Kouznetsova, 2007).

3. EXISTING TECHNIQUES

The main tool for programming in BlueJ is the code editor. The code editor displays the source code for the class. BlueJ projects, like standard Java packages, are directories containing the files included in the project. The source code of a class is the code that specifies all of the properties and characteristics of that class and its objects. The programmer can command the objects in his scenario to perform tasks or answer questions by writing source code, or syntax, in the Java programming language. When selecting the Open Editor from the class's menu to see the editor window that contains the class's source code. The source code displayed defines what the objects of the class can do (Runceanu, 2014).

4. OUR CONTRIBUTIONS

For course Object-Oriented Programming (OOP) we developed some applications in Java, with IDE BlueJ, for helping students to understand object-oriented programming techniques. We try to present a lot of programming techniques: Recursion, Lists, Stacks and Queues, Divide et Impera, Backtracking, Greedy and others. Divide and conquer method has many practical applications. One problem is known as the Towers of Hanoi. In brief, method consist in:

We consider three towers numbered A, B, C and n perforated disks having different diameters.

Initially all disks are placed on the tower A, in ascending order of their diameters, considering the direction of the top of the tower at its base.

To move all the disks on tower B in the same order using Tower C and the following rules:

1. At each step moves one disk
2. Continuously on each individual turn disc may occur only above the smaller diameter discs.

Solving this problem is based on the following considerations logic:

- if n = 1, then A -> B is immediate move (move disc from A to B)
- if n = 2, then the sequence moves is: A -> C, A -> B, C -> B
- if n > 2 the following:
- Move (n-1) disks A -> C
- Moving a disc A -> B
- Move the (n-1) disks C -> B

Note that the initial problem is decomposed into three simpler subproblems similar initial problem:

– move (n-1) disks A -> C

– move the last disk B

– move the (n-1) disks C -> B

The size of these sub-problems are: 1 n-1, n-1. These subproblems are independent because the initial tower (which are arranged discs) final tower and the tower intermediate are different.

We make a note: H (n, A, B, C) = n moves from tower A to tower B, using tower C

For

n = 1, we make move A -> B

n > 1, we call recursive function H (n, A, B, C) = H (n-1, A, C, B), A -> B, H (n-1, C, B, A)

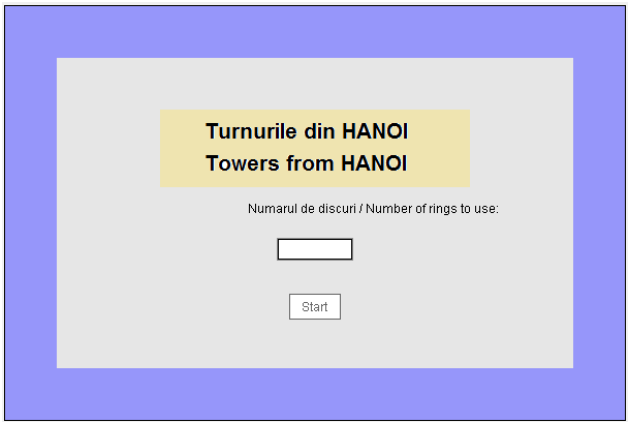

Fig. 5 - Recursion with BlueJ Towers of Hanoi Game

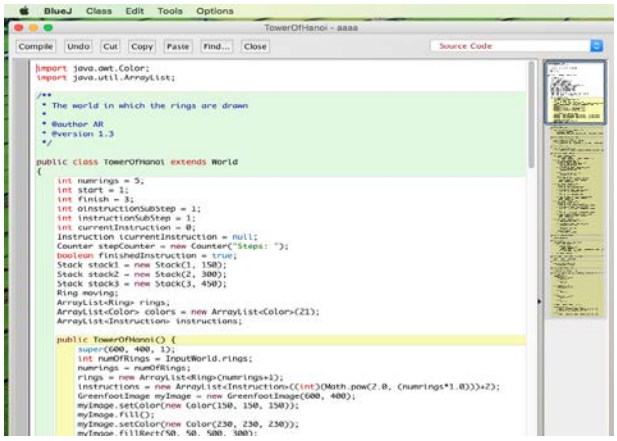

Fig. 6 - Source Code for the Towers of Hanoi Game

The game explains the movements required to transfer be made to move an entered number of discs from the first tower to the last, using as an intermediate tower the middle one. The player can move only a disc at a time, and a disc cannot be placed on top of a smaller one. Students can play the game varying the input data, which is the number of discs (Fig. 5).

Once the game has been played, student is requested to edit the Towers class to see the recursive function used (Fig. 6). The variables origin, destination and auxiliary correspond to the first, third and second tower respectively, and the n contains the entered number of discs for that execution.

Students can now understand how a recursive function makes call to itself, and what are the parameters on every call.

CONCLUSION

In this article it was presented the advantages of using a visual development environment. BlueJ is a visual development environment that help beginning students interact with classes and objects, methods to check behavior during execution and debug applications written code. It was use recursion applications for exemplifying implementation classes and objects in the course of Object Oriented Programming.

REFERENCES

Baldwin, L. and Kuljis, J. (2000). Visualisation Techniques for Learning and Teaching Programming, *Journal of Computing and Information Technology*, vol. 8, no. 4, pp. 285–291.

Barnes, D.J., Kolling, M. (2003). Objects First with Java. A Practical Introduction Using BlueJ, Harlow: Pearson.

BlueJ, http://www.bluej.org/

Hagan, D. and Markham, S. (2000a). Does it help to have some programming experience before beginning a computing degree program? *Proceedings of Integrating Technology into Computer Science Education Conference ITiCSE-2000*, pp. 25-28.

Hagan, D. and Markham, S. (2000b). Teaching Java with the BlueJ environment. *Proceedings of Australasian Society for Computers in Learning in Tertiary Education Conference ASCILITE 2000*.

Kouznetsova, S. (2007). Using BlueJ and Blackjack to teach object-oriented design concepts in CS1. *Journal of Computing Sciences in Colleges*, vol. 22, no 4, pp. 49 – 56.

Runceanu A. (2014). An implementation in Greenfoot used in teaching programming techniques, *Annals of the "Constantin Brancusi" University from Targu-Jiu, Engineering Series*, vol. 2, pp. 33-42.

ANNALS OF THE UNIVERSITY OF CRAIOVA
Series: Automation, Computers, Electronics and Mechatronics, Vol. 12 (39), No. 1, 2015

Running Complex Queries on a Graph Database: A Performance Evaluation of Neo4j

Călin Constantinov, Mihai L. Mocanu, Cosmin M. Poteraş

Department of Computers and Information Technology, University of Craiova, Romania (e-mail: constantinov.calin@ucv.ro, mmocanu@software.ucv.ro, cpoteras@software.ucv.ro).

Abstract: Computer science, by far one of the most dynamic research domains, manages to produce concepts, prototypes and paradigms at a pace that is sometimes hard to follow. Although most of these ideas set goals that are more or less realistic, some of them are able to fascinate by their potential to lead to dramatic changes. One particular case is represented today by graph databases, a type of NOSQL solution which can be applied to very power demanding tasks in data analysis frameworks. They allow for the expansion of data sources, including social media feeds, thus making complex systems such as recommendation engines much more powerful. No matter the controversies among both simple technology enthusiasts and large corporations, these databases are rapidly expanding, managing to not only generate interest, but also remarkable solutions. In this paper, we will have a deep look over the most popular graph database, Neo4j, evaluating its performance and scalability options when running very complex statistics and recommendation queries over a dataset of a significant size, containing strongly interconnected Facebook data. The evolution from a simple standalone database scenario to a Highly-Available (HA) cluster setup is described step by step by analysing the impact that each configuration change has on the system's both read and write performance. Given the positive results of this paper, we believe that graph technologies represent a very promising research domain as, considering their performance, they are likely to hold the key to building an efficient, distributed social-network graph mining framework on which data analysis jobs can be continuously run, further enhancing the data.

Keywords: Graph Database, Facebook Data, Performance Evaluation, Social Recommendation Engine, Data Analysis

1. INTRODUCTION

Although NOSQL databases faced initial reticence from developers used to the comfort and safety inspired by SQL, we are now seeing a strong raise in the popularity of some of these novel persistence solutions, as most of the large corporations are currently at least experimenting with them. Apart from the promising performance gain, some products are reaching maturity and many of the NOSQL drawbacks are being rapidly dealt with, further increasing interest. Although SQL databases still excel in many situations, the need to store very large amounts of data has revealed the difficultly of scaling these traditional solutions. Moreover, given the large number of relationships between entities, modern data is starting to have a graph-like structure. Unfortunately, SQL does not naturally support graph specific operations such as finding the shortest path between two nodes. Complex stored procedures and queries are thus needed for even the simplest tasks.

For instance, given its popularity, Facebook always represents an interesting case-study. Over the years, this platform has generated a lot of interest and has been taken as a reference for all social networks. The rate of growth has been remarkable: not only has the network seen a steep rise in the number of total active users, but also the quantity and complexity of the information that these users can share and which needs to be processed has increased. Also considering that a typical user can have a triple digit number of friends, it is easy to imagine that the data stored by Facebook is not only large and diverse, but also very strongly connected, as mentioned in paper Bronson et al. (2013). Basically, a very large portion of the world's active population is present on Facebook, creating, consuming and sharing content. This is why a lot of businesses have transited from promoting products using a website to simply managing a Facebook page, maximizing reach, strengthening relationships with clients and minimizing operational costs. The advantage of having all these kind of activities in the same place is that most of the information is semi-structured and can be easily mined and processed, given that the right tools are deployed for the job.

The need to process all this data in a reasonable amount of time has led to the evaluation of SQL alternatives. One of the most promising of them is represented by graph databases, as they naturally model social data. In this paper we aim to simulate very complex data analysis jobs

by performing socially-enhanced recommendations on a large data set, a common task in the Web 2.0 world.

Traditional recommender systems have been used by companies or shops that sell an item or a service in the attempt to cross-sell an additional product, as described by work Balabanović and Shoham (1997). There are usually a number of techniques implied: for example, if a certain product is frequently bought together with another product, it makes sense to advertise the second product along with the first one to the potential buyer. Another example would be to identify pairs of users that typically buy the same products and recommend products that just one of them has bought to the other one. Furthermore, collaborative filtering was used for predicting the way in which a certain user would evaluate a product or service, based on his similarity to other users. Modern day engines however have much more social data available for taking into consideration before performing a recommendation. The greatest challenge is to be able to run recommendation queries fast, without significant performance penalties as the size of the database grows. This is difficult to achieve using SQL, but can be attempted using a graph database such as Neo4j which promises easier scalability.

2. BACKGROUND

As mentioned in article Angles and Gutierrez (2008), alternatives for relational databases started to appear as early as the beginning of the nineties when object-oriented databases were developed for several data-intensive applications. Roughly in the same period, the first graph models were developed, but were gradually abandoned until recently. Simple record-type data having a fixed schema is no longer typical for modern applications as this can represent a major drawback in an era flooded by semi-structured information.

The need to store and process data of graph-like nature has, however, revived graph databases over the last few years, providing developers with powerful tools for capturing domain semantics within a visual data model. It is mentioned in the work that nowadays information interconnectivity and topology is at least as important as the data itself, the true value being represented by the links between entities. Moreover, contrary to SQL for which exploring the underlying web of relationships is a difficult task, this new database model comes with a much more intuitive querying language paradigm.

In the last few years, many graph database implementations have been released and choosing the right one is not always an easy task, as they also tend to evolve at a fast pace. Fortunately, many papers which try to benchmark and compare the alternatives are now available.

For instance, paper Ciglan et al. (2012) mentions that the graph data structure is an attractive abstraction for modelling real-world phenomena and insists on the importance of fairly assessing the performance of graph databases. Typically, performance is evaluated by looking over how well these databases can handle various traversal operations as this is what queries come down to in a graph. Two sets of tests are conducted, which match the types of queries that we have run over the system that we will be describing in this paper: one dealing with pure query-like traversals which have random starting points and search for related vertices and a seconds aiming to simulate whole graph traversal operations typically used more complex jobs such as computation of connected components analysis or centrality measures. In this preliminary experiment, Neo4j was tested along alternative databases, obtaining satisfactory results.

The ideal approach for traversal operations is to have the whole graph structure cached in the volatile memory as these operations are characterised by random memory accesses, which are known to be very time consuming for a persistent storage. However, as it will be later detailed, there are many cases for which the whole graph does not fit in the main memory. In such situations, some optimisations can be attempted in order to have most of the queried sub-graph cached.

Work Jouili and Vansteenberghe (2013) states that more and more companies are starting to provide services which can no longer be efficiently managed by using traditional relational databases, forcing them to seek alternative technologies. Graph databases are again mentioned as a possible solution for many types of applications and the paper aims to evaluate a number of implementations from a client's side perspective: measuring the time that a client has to wait between issuing a request and receiving a response from the database, thus including communications overhead. Neo4j is concluded to have the best overall performance, while standing out for its predictable behaviour.

As another novelty, the benchmark introduced in the work supports simulating multiple concurrent clients located on a number of distributed machines. This is something similar to what is happening in the application that we have developed.

Reference Holzschuher and Peinl (2013) provides an insight on Cypher, Neo4j's graph query language, concluding, based on performance and ease of use, that it is a promising candidate for becoming a standard. The authors experiment with a Web 2.0 inspired set of data for which it is now common to use NOSQL alternatives. Modelling this information in a relational database causes a high number of many-to-many relationships which in turn leads to a succession of very costly JOIN operations when querying the data. Graph databases are again recommended for these use-cases and an observation is made about the fact that these technologies can be placed somewhere between the SQL and NOSQL worlds as they are not simple aggregate solutions (such as Key-Value Stores, Column-Family Stores or Document Databases) and usual favour consistency and availability as opposed to partition-tolerance.

The study compares Cypher with Gremlin, an alternative graph query language which, although outperforming the former in a number of scenarios, is not preferred because of having the disadvantage of not being as maintainable and as readable. SQL is partially compared with Cypher and, as expected, performs much worse. Moreover, the experiment confirms that growing the number of entities stored in the graph database does not significantly affect performance as it does in case of an SQL solution. The reason for this is that only the local neighbourhood of a given node is traversed, no matter the size of the

whole dataset. The paper however states that it is not over-stressing highly graph-related queries such as group recommendation queries, which is something that our experiment will focus more on.

An even more thorough comparison between a graph and a relational database can be a can be found in article Batra and Tyagi (2012). Once again, it is noted that when storing social data that is expected to evolve, SQL is not an optimal approach. Although Neo4j is mentioned as not being a mature solution, it provides an easily mutable schema and outperforms MySQL in all the experiments carried out, also showing that the performance gain raises as the size of the database grows.

Scalability is further analysed in work Dominguez-Sal et al. (2010) where four of the most popular graph databases are evaluated using the HPC Scalable Graph Analysis Benchmark using a set of generated data. The tests range from measuring the edge and node insertion along with the creation of initial indexes performance to evaluating the time needed to traverse the whole graph when computing the Betweenness Centrality indicator for certain nodes. Neo4j and DEX, an alternative graph database, were able to achieve the best performance, having no problems scaling up to a dataset consisting of 1 million nodes. Additionally, the importance of having implemented an optimal caching strategy for graphs that do not fit in the main memory is also mentioned in the paper.

It is important to state that all the referenced papers work with synthetic datasets as compared to our tests which were run over real Facebook data. Moreover, older Neo4j versions are used and, as also anticipated by these studies, some of the problems identified are claimed to have been addressed in the newer releases.

Additionally, when dealing with very large datasets, horizontal scaling seems to be the way to go for increasing performance. However, an issue faced by engineers is the fact that although processing power has seen great increases in the last decades, network transfer rates fall way behind. This is why, when trying to improve parallel performance, it is important to try to limit or optimize data transfers over the network. Thus, it makes more sense to try to migrate the computational task to the data than to bring the data where the computational task is being executed. This technique is called computational steering, which our previous work Poteraş et al. (2011) describes in greater detail. A brief discussion on what can be attempted in case of our system will be detailed in a later section.

3. IMPLEMENTATION

Neo4j is currently the most popular graph database, reason for which we chose it for our implementation.

This database can be deployed in two ways:

(1) **Standalone server**: meaning that just one instance of the database is used. Two configurations are possible:
   (a) *Server mode* - as typical for any traditional database, in this configuration Neo4j runs independently of the main application. For communication, a set of REST services are exposed.
   (b) *Embedded mode* - although it only works with applications that run in a Java Virtual Machine, in this configuration Neo4j is tightly coupled to the main application. As it can be expected and also seen in our experiments, very high performance can sometimes be obtained in this scenario. However, there are also many disadvantages when sharing the volatile memory between the application and the database.

(2) **Highly-Available cluster**: allows for a number of database instances to be used together in order to improve performance. A Neo4j HA cluster can be composed of both Server-mode running Neo4j instances as well as Embedded-mode running ones, each of them operating on a different machine. Communication between instances is realized over a custom protocol that tries to eliminate as much overhead as possible. A Neo4j HA cluster has of a single master node and a number of slave nodes. By only allowing write operations to be performed through the master (even if a write request is issued to a slave node, the master will eventually be called to instrument the operation), ACID transactions can be imposed. In order to complete a write operation and guarantee consistency, at least N / 2 + 1 servers from the cluster need to be available (out of N, the total number of servers in the cluster). If this quorum is not met, the cluster functions in read-only mode. As expected, write performance can sometimes be worse on a cluster then on a single database instance, due to the cluster management overhead. Neo4j's major downside is that each node instance has a copy of the whole database, so as for now, the database does not support graph sharding. Graph sharding is a NP-complete problem which is even more difficult to solve considering that the graph is constantly evolving. Read performance can typically see near linear speedup in a cluster configuration, as long as the whole graph can be stored in-memory. Even if this is not the case, it can be assumed that, for general cases, it is faster to load portions of the missing graph from a local permanent storage device than to call for them over the network.

In a Neo4j Highly-Available cluster setup, it is typical for a Java application to issue write requests to an Embedded-Master node, while the read requests to be directed to a Server-Slave node. In case there are multiple Server-Slave nodes, a load balancer such as HAProxy can be used. An example for this setup will be detailed in a later section.

But this kind of configuration can be the starting point for a much more sophisticated graph database cluster. Although social graph sharding might be very difficult to achieve at this moment, cache sharding remains a valid approach for improving read times. In cases were the graph is very large, it might be impossible to store it entirely in the very fast, but expensive, volatile memory. This is especially true for Neo4j which employs a native graph storage. However, Neo4j's cache is filled with portions of the graph most heavily accessed, so by using an advanced load-balancing algorithm, requests could be routed to an instance of Neo4j likely that holds most of the queried graph in its cache. The trick would be to identify usage

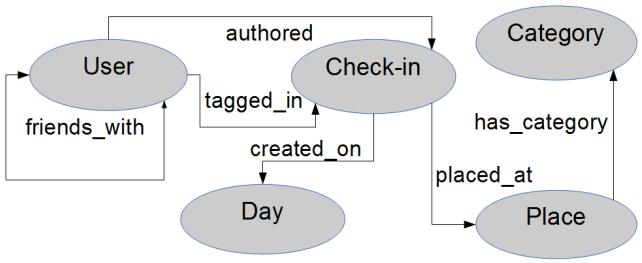

Fig. 1. Neo4j Data Model

patterns by means of statistical analysis and customize the HAProxy load balancer to forward requests to the ideal instance of the graph database. In a way, this is also a form of computational steering, as described in a previous section. This is an active research domain which could lead to building an optimal distributed social-network analysis framework, but further discussions are outside the scope of this paper.

The application aims to constantly mine Facebook information and gather as many check-ins placed in the city of Craiova as possible and use them for performing recommendations and computing statistics. As Facebook data is not generally public, user authorization tokens are required. Thus, two main modules were developed:

(1) **Core Web Module**: using Facebook Graph API 1.0 and Facebook Query Language, check-ins are gathered and stored in Neo4j. The graph data model can be seen in Fig. 1. The module also exposes a REST service that will be called by the user-facing module during Facebook login. This service saves the user's Facebook authorization token in the databases which will be then used for further gathering Facebook data. All database write requests are issued by this module. In order to guarantee that no duplicate data is stored, the module also sends a number of read requests.
(2) **User MVC Module**: a number of statistics and recommendations that are detailed bellow are available to a logged in user. Initially, a user has to login and authorise the application using his Facebook account. This module only sends read requests to the database.

A number of basic recommendations and statistics were written using Cypher and were made available to the users:

(1) **Global statistics and recommendations**:
   (a) *Total number of users / places / check-ins* - simple statistics counting the number of particular items stored in the database.
   (b) *Most popular categories* - for example: Bar, Pub, Italian Restaurant.
   (c) *Most popular places, by the number of check-ins* - the number of check-ins for each place is counted.
   (d) *Most popular places, by number of visitors* - some visitors place more than one check-in in the same location, so the number of unique visitors is counted.
   (e) *Most popular places, by the percentage of visitors that have returned at least once* - for each place, out of the total number of visitors, the percentage of visitors that has checked-in at the given location at least twice is counted.
(2) **Recommendations and statistics given a specific user**:
   (a) *Friends with the most check-ins* - the number of check-ins of each of the current user's friends is counted.
   (b) *Suggestions for new friends* - non-friends most often tagged in the same check-in as the current user are identified.
   (c) *Suggestions for new places to visit* - non-visited locations most often visited by the current user's close friends are identified. Close friends are identified by counting the number of check-ins in which both the friend and the given user are tagged in.
(3) **Recommendations and statistics given a specific place**:
   (a) *Similar places, based on their common visitors* - the number of common visitors the given place has with every other place is counted.
   (b) *Similar places, based on their common categories* - the number of common categories the given place has with every other place is counted. In case of ties, the number of check-ins placed by common visitors (as described previously) is also counted for every other place.

As it can be observed, for a number of the above queries, a graph-wide traversal will need to be performed.

For developing our platform, the following technologies were used: Java SE 7, Spring Data Neo4j 2.3.4, HAProxy 1.4.25 and Neo4j 2.0.3.

4. EXPERIMENTAL RESULTS

All tests were run 3 times for 25 minutes and started from the same dataset consisting of 21981 users, 48051 check-ins, 549 places and 76 categories, all linked by 392607 relationships. During each test, data from Facebook was further retrieved by 16 threads that operate by also making sure no duplicate information is stored in Neo4j. This means that, before saving new data, a worker thread needs to issue both read and write requests which will be considered as a whole and will be called a data persisting operation. For simulating client usage, 96 threads were constantly sending random statistics or recommendation requests (form the previously described list) to the system. The processing times presented where measured inside the main application, thus only considering network overhead inside the cluster (where applicable).

4.1 Neo4j Server vs Neo4j Embedded

This first performance comparison was conducted by using a single computer (to be further referenced as **PC1**), having the following configuration: 4th Gen Intel i7 @ 2.2 GHz, 8 GB DDR3 RAM @ 1600 MHz and HDD @ 5400 RPM. The size of our database is small enough for Neo4j to cache all of it in the computer's RAM memory, so the impact of a slower HDD is minimized for read operations. The results can be seen in Fig. 2.

The first test was run using Neo4j configured in a classic Server configuration, exposing a set of REST endpoints. For eliminating network overhead, the server was run on

Run	Time (ms) Data persist (write)			Run	Time (ms) Recommendations (read)		
	Server	Embedded			Server	Embedded	
1	9157	1027		1	991	1166	
2	10789	1039		2	998	1362	
3	11530	1267	Gain	3	812	1561	Gain
Average:	**10492**	**1111**	**+ 844 %**	**Average:**	**933**	**1363**	**- 31%**

Fig. 2. Neo4j Server vs Neo4j Embedded

the same machine as the application. The mean time for finalizing a write request (from the moment the application receives a data response from Facebook and up to the moment when the data is successfully committed to the database) was disappointing: 10492 milliseconds. Using a high level Neo4j REST API along with the HTTP overhead of each request greatly affects performance.

On the other hand, a very promising result of just 993 milliseconds for completing a recommendation request was obtained (with a very small percentage of worst performing queries timed just slightly above 2100 milliseconds), suggesting that the system is quite capable of handling a large number of traversals. However, because of the slow performing write operations, the graph did not suffer dramatic structural changes at a high pace, thus not forcing Neo4j to invalidate and refresh the content of its cache, which, in turn, makes high performing read operations possible.

A first improvement that we brought to the system was to configure Neo4j in Embedded mode and run it in the same Java Virtual Machine as our application. This allows Spring Data Neo4j to use low level (and thus fast) Neo4j Core API functions when performing write operations, as detailed in Hunger (2012). Data persisting operations now took only 1111 milliseconds resulting in an impressive 844% performance gain. However, recommendation

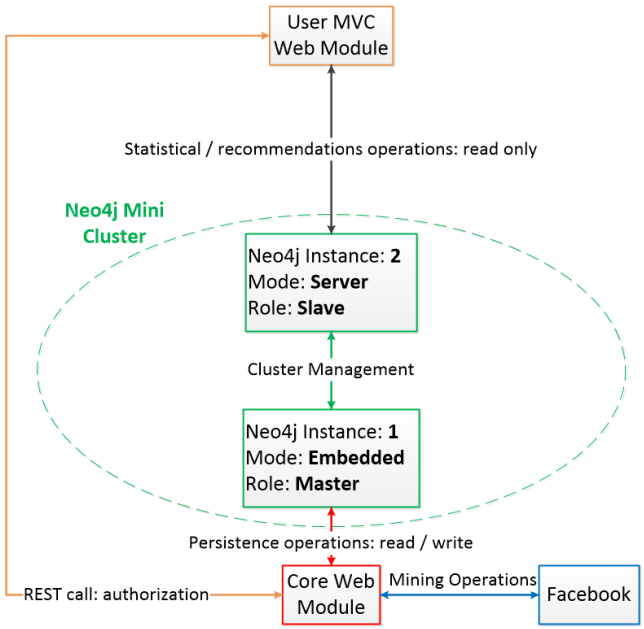

Fig. 3. Neo4j Mini-Cluster

requests now took an average of 1363 milliseconds to complete, meaning a 31% performance drop. As previously explained, because of the much higher number of write operations, this was to be expected.

4.2 Neo4j Embedded vs Neo4j Mini-Cluster

For computing recommendations that have a high probability of being relevant, a lot of data needs to be processed. This data also needs to always be up to date, so it is important for our system to quickly store new information. Although we managed to improve write times using the previous configuration, we unfortunately ended up increasing recommendation times. We decided to try and scale our system by adding an additional computer (to be further referenced as **PC2**), having the following configuration: Intel Core2Quad @ 2.83 GHz, 4 GB DDR2 RAM @ 1066 MHz and HDD @ 7200 RPM. The configuration can be seen in Fig. 3. The results can be seen in Fig. 4.

PC1 remained configured as before, with an Embedded Neo4j instance (now having the role of a HA cluster Master node) while another Neo4j Server instance was deployed as a Slave node over a Gigabit LAN on **PC2**. All recommendation requests were now sent to **PC2** that only had to handle read operations (queries). All previous tests were run again and the newly obtained results were analysed. For writing operations, a mean time of 841 milliseconds was observed, accounting for a 32% performance gain. Although **PC1** now only had to handle data persistency requests, the result is still surprising because this machine also needed to coordinate the HA database cluster. This in turn means that all data written on the Embedded instance of Neo4j needed to be replicated on the Server instance. One reason for obtaining this result is the fact that the two database instances communicate through a custom protocol that does not have as much overhead as REST calls over HTTP.

For recommendation operations, a mean time of 1186 milliseconds was obtained, meaning a 15% performance increase. However, this result is still 21% behind the result managed by our first configuration (for which, however, graph structure changes were not as frequent). Considering all the above aspects, we can conclude that, overall, this was the most performant setup of the three configurations analysed so far.

4.3 Neo4j Mini-Cluster vs Neo4j HAProxy Cluster

In attempt to further improve read operation times, we decided to enhance our mini-cluster by adding one more

Run	Time (ms) Data persist (write)			Run	Time (ms) Recommendations (read)		
	Embedded	Mini-Cluster			Embedded	Mini-Cluster	
1	1027	934		1	1166	1029	
2	1039	840		2	1362	1236	
3	1267	751	Gain	3	1561	1294	Gain
Average:	**1111**	**841**	**+ 32 %**	**Average:**	**1363**	**1186**	**+ 15%**

Fig. 4. Neo4j Embedded vs Neo4j Mini-Cluster

machine to it (to further referenced as **PC3**). This machine had the following configuration: Intel Core2Duo @ 3.16 GHz, 4 GB DDR2 RAM @ 1066 MHz and HDD @ 7200 RPM. The configuration can be seen in Fig. 5. The results can be seen in Fig. 6.

PC3 was used for running an additional Neo4j Server Slave node connected to the cluster. A HAProxy load balancer was installed on **PC2** and was used for forwarding all recommendation requests to a Slave node running on either **PC2** or **PC3**.

The tests were executed one last time, leading to the following results: for the write operations, an average execution time of 829 milliseconds was obtained, accounting for a 1% gain (which could reasonably be considered within an error margin). What is important to note is the fact that the performance of the internal communications protocol within the cluster for the Master node is not significantly impacted by the addition of an extra Slave node. Considering that the version of Neo4j used for our experiments does not support parallel write operations, it can be concluded that the only way to improve execution times would be to vertically scale **PC1**.

For recommendation requests, the new configuration managed to achieve a 38% performance gain, finalizing, on average, each request in 857 milliseconds (accounting for a

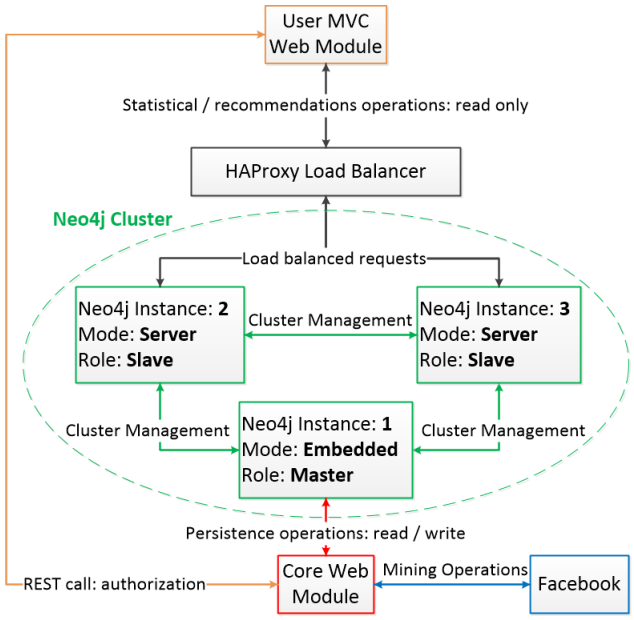

Fig. 5. Neo4j HAProxy Cluster

9% gain compared to the initial stand-alone server setup, although, as it was pointed out, this is not necessarily relevant as the database also handles much more structural changes in this final configuration). While the results are far from showing a linear performance improvement, they are still remarkable. In order for the Slave nodes to constantly catch-up with the rest of the cluster, somewhat significant computational times on **PC2** and **PC3** are used by the cluster consistency synchronization mechanisms. Near linear scaling is probably to be expected for a scenario where no data is written to the cluster. Moreover, the HAProxy load balancer was running on **PC2** meaning that the overhead for forwarding requests over the network was eliminated for this machine. Lastly, **PC3** is somewhat less powerful than **PC2**.

5. CONCLUSIONS AND FUTURE WORK

Although the initial results obtained for writing data to an ordinary Server Neo4 instance were disappointing, by progressively enhancing the system configuration, performance was greatly improved. In the end, it was shown that execution times for complex queries are quite reasonable and that the system can also be horizontally scaled without considerable effort.

Unfortunately, write operations remain the bottleneck of any Neo4j cluster configuration so future efforts could be taken in order to optimize them. An idea for a real-world application would be to store and aggregate all write requests over a period of time and perform all the operations when low incoming read request rates are detected. This is only valid for cases where postponing the write operations doesn't have a serious impact on the application.

As already mentioned, NOSQL solutions evolve very quickly. Because of this, we plan to successively upgrade to major versions of Neo4j from the last couple of years and run the initial tests again in order to assess the performance improvements of each version for our specific use-case. Moreover, we are going to further improve the cluster by adding an additional machine that will work as a dedicated load-balancer.

As query execution times were very low, as a long term focus area, the system can be used as a starting point for attempting to build a complex data analysis platform that would try and identify clusters of strongly related nodes along with their most representative entities, based on the interactions within each node cluster. Moreover, it would make sense to use this tightly interconnected clusters for

Run	Time (ms) Data persist (write)			Run	Time (ms) Recommendations (read)		
	Mini-Cluter	HAProxy Cluster			Mini-Cluster	HAProxy Cluster	
1	934	826		1	1029	986	
2	840	910		2	1236	866	
3	751	753	Gain	3	1294	721	Gain
Average:	**841**	**829**	**+ 1 %**	**Average:**	**1186**	**857**	**+ 38%**

Fig. 6. Neo4j Mini-Cluster vs Neo4j HAProxy Cluster

achieving basic cache sharding, by optimally choosing the slave instance to which to redirect requests, as previously described.

As an example, it can be observed that the friendship relationships are not annotated in any way, although it is clear that in any group of users some establish themselves as influencers while others become simple followers. These relationships can be constantly analysed and weighted. Thus communities along with their respective opinion leaders can be identified. Such information can be proven to be valuable as it would allow us to also weight individuals and their impact. This data can be used, for instance, by recommendation engines to further improve the relevance of their computed suggestions.

REFERENCES

Angles, R. and Gutierrez, C. (2008). Survey of graph database models. *ACM Comput. Surv.*, 40(1), 1:1–1:39. doi:10.1145/1322432.1322433. URL `http://doi.acm.org/10.1145/1322432.1322433`.

Balabanović, M. and Shoham, Y. (1997). Fab: Content-based, collaborative recommendation. *Commun. ACM*, 40(3), 66–72. doi:10.1145/245108.245124. URL `http://doi.acm.org/10.1145/245108.245124`.

Batra, S. and Tyagi, C. (2012). Comparative analysis of relational and graph databases. *International Journal of Soft Computing and Engineering (IJSCE)*, 2(2).

Bronson, N., Amsden, Z., Cabrera, G., Chakka, P., Dimov, P., Ding, H., Ferris, J., Giardullo, A., Kulkarni, S., Li, H., Marchukov, M., Petrov, D., Puzar, L., Song, Y.J., and Venkataramani, V. (2013). Tao: Facebook's distributed data store for the social graph. In *Presented as part of the 2013 USENIX Annual Technical Conference (USENIX ATC 13)*, 49–60. USENIX, San Jose, CA. URL `https://www.usenix.org/conference/atc13`.

Ciglan, M., Averbuch, A., and Hluchy, L. (2012). Benchmarking traversal operations over graph databases. In *Data Engineering Workshops (ICDEW), 2012 IEEE 28th International Conference on*, 186–189. doi: 10.1109/ICDEW.2012.47.

Dominguez-Sal, D., Urbón-Bayes, P., Giménez-Vañó, A., Gómez-Villamor, S., Martínez-Bazán, N., and Larriba-Pey, J.L. (2010). Survey of graph database performance on the hpc scalable graph analysis benchmark. In *Proceedings of the 2010 International Conference on Web-age Information Management*, WAIM'10, 37–48. Springer-Verlag, Berlin, Heidelberg.

Holzschuher, F. and Peinl, R. (2013). Performance of graph query languages: Comparison of cypher, gremlin and native access in neo4j. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, EDBT '13, 195–204. ACM, New York, NY, USA. doi:10.1145/2457317.2457351. URL `http://doi.acm.org/10.1145/2457317.2457351`.

Hunger, M. (2012). *Good Relationships: The Spring Data NEO4J Guide Book*. InfoQ enterprise software development series. C4Media.

Jouili, S. and Vansteenberghe, V. (2013). An empirical comparison of graph databases. In *Social Computing (SocialCom), 2013 International Conference on*, 708–715. doi:10.1109/SocialCom.2013.106.

Poteraş, C., Constantinov, C., and Mocanu, M. (2011). The evolutionary design of a framework for computational steering. *Annals of the University of Craiova*, 8(2), 50–59. URL `http://ace.ucv.ro/anale/content2011vol8nr2.html`.

Author Index