Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

RESEARCH ARTICLE

Constraint Reformulations for Set Point Optimization Problems using Fuzzy Cognitive Map Models

A. Garzón Casado[1] | P. Cano Marchal[1] | C. Wagner[2] | J. Gómez Ortega[1] | J. Gámez García[1]

[1]Robotics, Automation and Computer Vision Group. University of Jaén. Agrifood Campus of International Excellence (ceiA3)
[2]Lab for Uncertainty in Data and Decision Making (LUCID), School of Computer Science. University of Nottingham

Correspondence
Pablo Cano Marchal. Email: pcano@ujaen.es

Summary

The selection of optimal set points is an important problem in modern process control. Fuzzy Cognitive Maps (FCMs) allow to construct models of complex processes using expert knowledge, which is particularly useful in situations where measuring the variables of interest online is problematic. These models can be used as constraints in optimization problems with the objective of determining optimal set points for those processes. This paper presents a reformulation of the constraints imposed by the FCM models that reduces the complexity of the resulting optimization problem and enables the application of heuristic methods for its solution. Computational results show that the use of separable programming on the reformulated problem constitutes a very good alternative, both in terms of solution time and reliability in finding the optimum, enabling the application of FCM modelling to larger systems and easing the practical implementation of the approach.

KEYWORDS:
Fuzzy Cognitive Maps, Decision Support Systems, Optimization, Hierarchical Control

1 | INTRODUCTION

The selection of the set points of process variables is a key step in the operation of modern process industries, as this decision has a high impact on the quality characteristics of the final product and on the process performance metrics. This decision usually requires addressing the process from a global point of view, since the optimal values for these set points usually emerge from a careful consideration of the trade-offs between conflicting operation objectives. The trade-off between final product quality and operation cost is the most common example of this situation.

Global process models, thus, are a fundamental tool to aid in the decision making process and are usually employed in the higher layers of multilayer control[1,2], whose purpose is precisely to provide these set points. The development of these models, however, is not always an easy task due to the complexity of the processes or difficulties in obtaining measurements of the involved variables. The food processing industry is a common example of this situation[3,4,5], as it is usual to have variables of interest related to quality features of the produced goods that are hard to measure online. In this scenario, a plausible approach for the development of models relating these variables with the rest of process variables is via expert operator knowledge.

Fuzzy Cognitive Maps (FCM)[6,7,8,9] are a convenient tool for modeling complex systems with many involved variables. They provide an intuitive representation of the relations among the variables, allow to easily decompose the model into simpler parts and to use an incremental approach in the construction of the models. These features render them as a very useful tool for the construction of models that are based on expert knowledge. FCM can also be used in circumstances where data is available; in this case, the models can be constructed using these data via optimization techniques. In fact, many research efforts have been devoted to the use of optimization techniques to find the parameters for the FCM models[10 11 12]. These methods are conceptually similar to system identification approaches, since the objective is precisely to obtain a model relating a set of input and output variables. The techniques employed, however, are typically different from the ones used by traditional system identification approaches, and can be roughly classified into three groups: Hebbian-based learning, population-based learning and mixtures of these two approaches. A very recent work has proposed a reformulation of this problem that casts it as a convex optimization problem[10], which significantly contributes to overcoming many of the shortcomings of previous approaches. The focus of this paper, however, is not the use of optimization techniques to obtain a model based on experimental data; but, assuming that a model is already available, to investigate the implications of the mathematical structure of the FCM models when they are used as constraints in optimization problems that aim to provide set points in an hierarchical control scheme, with a focus on aspects that influence the practical use of the approach.

The use of FCM plus an optimization problem to provide and update global process set points was explored in Ref.[13] with an application example of the methodology to an industrial food manufacturing process where the resulting nonconvex optimization problem was initially solved using a general purpose global solver. However, this solution method is slow and viable only for relatively small models. Solving optimization problems efficiently is a topic of clear practical interest, as demonstrated by recent research effort on the topic[14,15,16]. Furthermore, the possibility of solving relatively large problems quickly could boost the application of this type of modelling to larger production planning problems, such as the one presented in[17].

This paper elaborates on the work presented in[13] and presents a reformulation of the constraints introduced by the FCM models that ease the complexity of the optimization problem and allows the use of simpler methods for its solution. This way, results of the application of convex-concave and separable programming to the reformulated problem are presented, showing that separable programming constitutes a very good alternative, both in terms of solution time and reliability in finding the optimum. The use of separable programming on the reformulated problem paves the way for the use of FCM models on larger systems, and enables the use of standard Mixed-Integer Linear Solvers for its solution, which eases the practical implementation of the approach. The remainder of the paper is organized as follows: Section 2 provides a brief overview of FCM and the heuristic solution methods. In turn, Section 3 introduces the reformulation of the optimization problem and Section 4 shows the computational results obtained in experiments comparing the two different heuristic solution methods. Finally, Section 5 presents the conclusions of the work.

2 | BACKGROUND

This Section provides a brief background on the modeling methodology and the solving methods for the nonconvex optimization problem.

2.1 | Fuzzy Cognitive Maps

Fuzzy Cognitive Maps were initially proposed in[6] and have evolved into many different variations of the original idea[8,9]. The modeling method employed in this work, detailed in[13], is composed of a set of nodes $\mathbf{V}$ that represent the process variables and a set of directed arcs $\mathbf{A}$ that symbolize the relationships between these variables.

For each node $v_i \in \mathbf{V}$ of the network, the following elements are defined:

- U_i: the universe of discourse of the node, i.e., the range of possible crisp values that the associated process variable may take.
- H_i: a collection of fuzzy sets (FS) L_i^k defined in U_i:

$$L_i^k = \{\langle t, \mu_{L_i^k}(t)\rangle \ : \ t \in U_i\}, \tag{1}$$

$$H_i = \{L_i^k,\ k = 1, 2, \cdots, K_i\}. \tag{2}$$

- y_i: the crisp value assigned to the node ($y_i \in \mathbb{R}$).
- $\mathbf{x}_i$: an array holding the membership value of y_i to each fuzzy set L_i ($\mathbf{x}_i \in \mathbb{R}^{K_i}$).

$$\mathbf{x}_i = [\mu_{L_i^1}(y_i)\ \mu_{L_i^2}(y_i)\ \cdots\ \mu_{L_i^{K_i}}(y_i)]^T. \tag{3}$$

In turn, the properties assigned to the arcs are:

- ρ_{ij}: intensity of the relation between the nodes v_i and v_j ($\rho_{ij} \in \mathbb{R}$).
- R_{ij}: a matrix that defines the relationships between the fuzzy sets of the nodes connected by the arc. This matrix is roughly equivalent to the rules in an Fuzzy Logic System (FLS). The size of this matrix is given by the cardinality of H_i, denoted K_i and the cardinality of H_j, denoted K_j. This way, $R_{ij} \in \mathbb{R}^{K_i \times K_j}$.

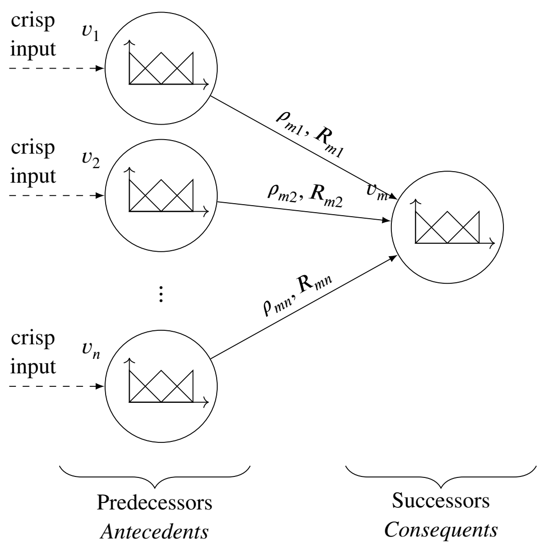

FIGURE 1 Generic multi-input single output model for traditional, singleton FCM. A more complex model can be built by composition of this elementary structure.

Figure (1) depicts a two-layer network. The tail nodes of the arcs $(v_1, v_2, \cdots, v_n)$ act as antecedents in a fuzzy rule, in the sense that their value is supposed to be known and influence the value of the head node. The head node, therefore, acts as the consequent part of a fuzzy rule, and its value is determined by the tail nodes and the properties of the arcs, according to the following algorithm:

1. For each tail node $v_1, v_2, \cdots, v_n$, the evaluation of the membership degree of y_i to each FS defined in U_i provides the entries of the corresponding vector $\mathbf{x}_i$:

$$\mathbf{x}_i = [\mu_{L_i^1}(y_i)\ \mu_{L_i^2}(y_i)\ \cdots\ \mu_{L_i^{K_i}}(y_i)]^T. \tag{4}$$

2. The impact received by the head node m is computed as:

$$\mathbf{w}_m = [w_m^1\ w_m^2\ \cdots\ w_m^{K_m}]^T = \sum_{j=1}^{n} \rho_{mj} R_{mj} \mathbf{x}_j. \tag{5}$$

3. The crisp value of the head node y_m is computed combining the peak values of the FS defined in v_m, denoted q_m^k, using $\mathbf{w}_m$ as weight (effectively, height defuzzification):

$$y_m = \frac{\sum_{k=1}^{K_m} w_m^k\ q_m^k}{\sum_{k=1}^{K_m} w_m^k}. \tag{6}$$

Multilayer models can be built by combining multiple simple models like the one depicted in Figure (1). In particular, constructing a multilayer model where v_m is also the predecessor of other nodes just requires considering the previously computed y_m as the crisp input to node v_m, and performing the steps 1 – 3 to compute the crisp value of the associated head node. More details on the methodology can be found in [13,18,19].

2.2 | Optimization Problem Solution Methods

Two approaches have been investigated for the solution of the optimization problem: Difference of Convex Programming[20] and Separable Programming[21]. The following Subsections briefly present the main features of both approaches.

2.2.1 | Convex-Concave Programming

Convex-Concave Programming is a heuristic method that provides local solutions to problems where nonconvex terms are expressed as differences of convex functions:

$$\begin{aligned} &\text{minimize } f_0(\mathbf{x}) - g_0(\mathbf{x}) \\ &\text{subject to } f_i(\mathbf{x}) - g_i(\mathbf{x}) \leq 0, \quad i = 1, ..., m. \end{aligned} \tag{7}$$

Here, $\mathbf{x} \in \mathbb{R}^n$ and f_j and g_j, $j = 0, 1, \cdots, m$ are all convex functions.

The main idea of the method is to iteratively solve convex approximations of the original problem obtained by linearizing the functions g_i, thus dropping the concave part of the problem:

$$\hat{g}_i(\mathbf{x}) = g_i(\mathbf{x}_k) + \nabla g_i(\mathbf{x}_k)^T(\mathbf{x} - \mathbf{x}_k). \tag{8}$$

The basic method requires a feasible starting point for the algorithm[22], however an extension of the method exists that introduces slack variables and a penalty on the violations of the constraints, and thus does not require an initial feasible point for the algorithm. Further details can be found in[20,23] and the references therein.

2.2.2 | Separable Programming

Separable programming deals with problems where the nonconvex constraints are *separable*, i.e., can be expressed as sums of functions of individual decision variables[21]:

$$f(\mathbf{x}) = \sum_i f_i(x_i).$$

As will be shown in Section 3, the nonconvex constraints in the optimization problem of interest can be expressed as separable constraints. This formulation has the advantange that each function causing a nonconvex constraint can be easily approximated by a piecewise linear function, yielding a program that can be solved as a Mixed-Integer Linear Program with a standard solver or as a Linear Program using a restricted entry simplex method[21]. In this paper, both inner and outer linearization methods were considered, and the results that were obtained using each of these methods were also compared. Figure 2 depicts the linearization of x^2 according to both methods.

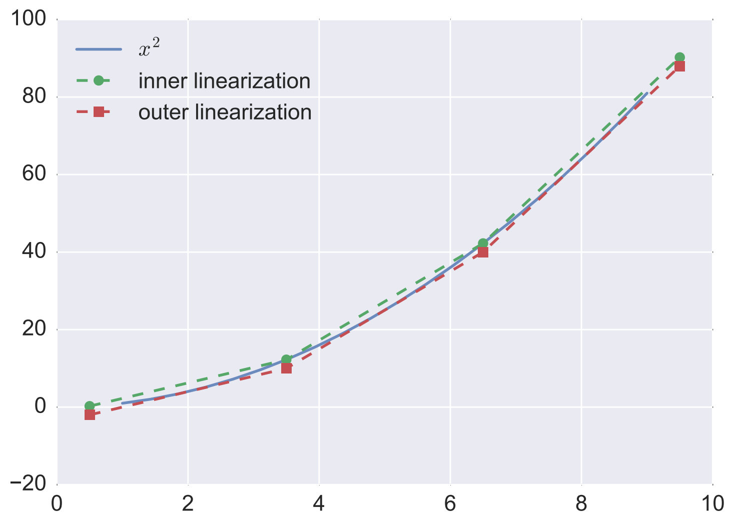

FIGURE 2 Inner and outer linearization of x^2. Both methods were employed in the piecewise-linear solution of the optimization problems.

3 | PROBLEM REFORMULATION

The formulation of the constraints of an optimization problem is known to have a considerable impact on the difficulty of its solution[24]. It is thus of interest to analyze the characteristics of Eqs. (4) – (6) when these are constraints of an optimization problem, and to consider alternative formulations that might reduce the complexity of the problem.

Equation (4) transforms y_i into $\mathbf{x}_i$ by means of the membership functions defined for each label. Typical membership functions are not linear – in fact, they are usually concave or pseudo-concave functions, e.g. triangular, trapezoidal or gaussian –, so expressing these constraints as equalities immediately yields the optimization problem as non-convex[25]. This equality constraint can be relaxed to an inequality if two additional linear constraints are introduced:

1. The weighted combination of the peaks of the FS (q_i^k) using the elements of $\mathbf{x}_i$, denoted x_i^k, must equal y_i:

$$y_i = \sum_{k=1}^{K_i} x_i^k \, q_i^k. \tag{9}$$

2. Since the FS are chosen to yield a fuzzy partition of the universe of discourse, the sum of the elements of $\mathbf{x}_i$ must equal 1:

$$\sum_{k=1}^{K_i} x_i^k = 1. \tag{10}$$

Introducing these constraints, Equation (4) can be relaxed to:

$$x_i^k \geq \mu_{L_i^k}(y_i). \tag{11}$$

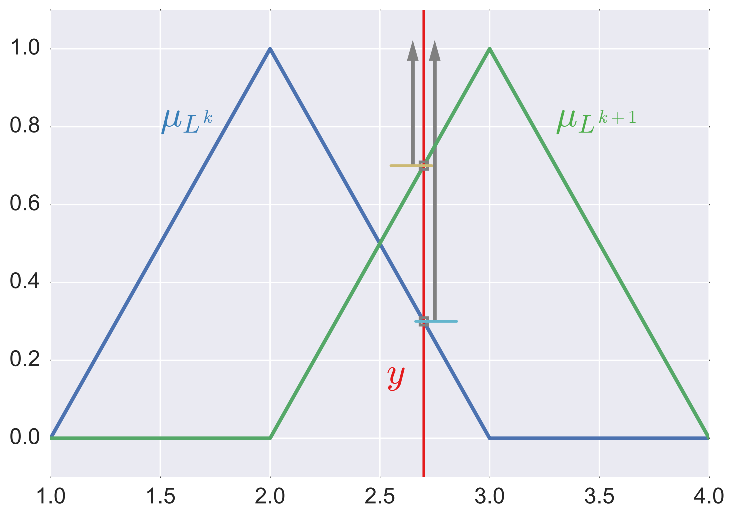

FIGURE 3 Triangular membership functions for two contiguous Fuzzy Sets. The arrows mark the sign of the inequality in Eq. (11).

Figure 3 shows two triangular membership functions (MF) for two generic FS defined in the universe of discourse of a node. According to Eq. (11), the membership value can be anything equal or greater than the membership value defined by the MF. However, since the MFs are chosen to yield a fuzzy partition and Eq. (10) applies, Eq. (11) is guaranteed to be satisfied as an equality. Unfortunately, the fact that $\mu_{L_i^k}(y_i)$ is typically a concave function still yields the constraint as non-convex. If triangular membership functions are used, this constraint can be modeled as a Special Ordered Set of type 2 (SOS2) constraint[26] on the elements of $\mathbf{x}_i$. It is easy to see that if the FS in H_i are defined to yield a fuzzy partition of U_i, then at most two consecutive elements must be nonzero, and the sum of all the elements of U_i must be one.

In turn, Eq. (5) provides K_i linear equality constraints that do not pose any complications for the solution of the optimization problem. On the other hand, Eq. (6) supplies a nonconvex equality constraint per node. In order to reformulate this constraint into a milder set of constraints, we define a variable c equal to the sum of the impact components:

$$c_i = \sum_{k=1}^{K_i} w_i^k. \tag{12}$$

Then, we can rewrite Eq. (6) as:

$$y_i\, c_i = \sum_{k=1}^{K_i} w_i^k\, q_i^k. \tag{13}$$

Once in this form, we can use the following standard reformulation trick to transform Eq. (13) into a linear constraint plus two separable quadratic equality constraints[21]. Two additional variables per node (a_i and b_i) are introduced and defined by the constraints:

$$\begin{aligned} a_i &= \frac{1}{2}(c_i + y_i), \\ b_i &= \frac{1}{2}(c_i - y_i). \end{aligned} \tag{14}$$

so that the following relation holds:

$$y_i\, c_i = a_i^2 - b_i^2. \tag{15}$$

We further introduce two extra variables (z_i and t_i), along with the constraints:

$$\begin{aligned} z_i &= a_i^2, \\ t_i &= b_i^2. \end{aligned} \tag{16}$$

Then, using Eq. (15) and Eq. (16), we can rewrite Eq. (13) as:

$$z_i - t_i = \sum_{k=1}^{K_i} w_i^k\, q_i^k, \tag{17}$$

which is a linear equality constraint. This way, the nonconvex constraint given by Eq. (6) has been recast into the nonconvex separable constraints defined by Eq. (16) and the three additional linear equality constraints contained in Eqs. (14) and (17).

4 | COMPUTATIONAL RESULTS

This Section presents a comparison of the performance of each of the considered solution methods for the reformulated problem, both in terms of reliability in finding the optimum and solution time. The experiments were carried out using the four models included in Figure 4, that were generated according to the following algorithm:

1. Choose (randomly or not):
   - Number of input and output nodes.
   - Total number of nodes and relations.
   - Number of layers and whether arcs may only influence nodes of their contiguous layer or not.
   - Frequency of the different type of relations
2. Randomly assign the free number of nodes into the intermediate layers of the model.
3. Connect each node in the network with a random node of the subsequent layer.
4. Connect isolated nodes with a random node of the previous layer.
5. Randomly connect nodes until the total number of relations is met, forcing them to be contiguous or not, as defined in step 1.

As shown in the Figure, models of different sizes were considered to evaluate the impact of model size in the solution time. For each model topology, four different assignments of relations and two cost allocations were considered, providing eight different optimization problems per topology.

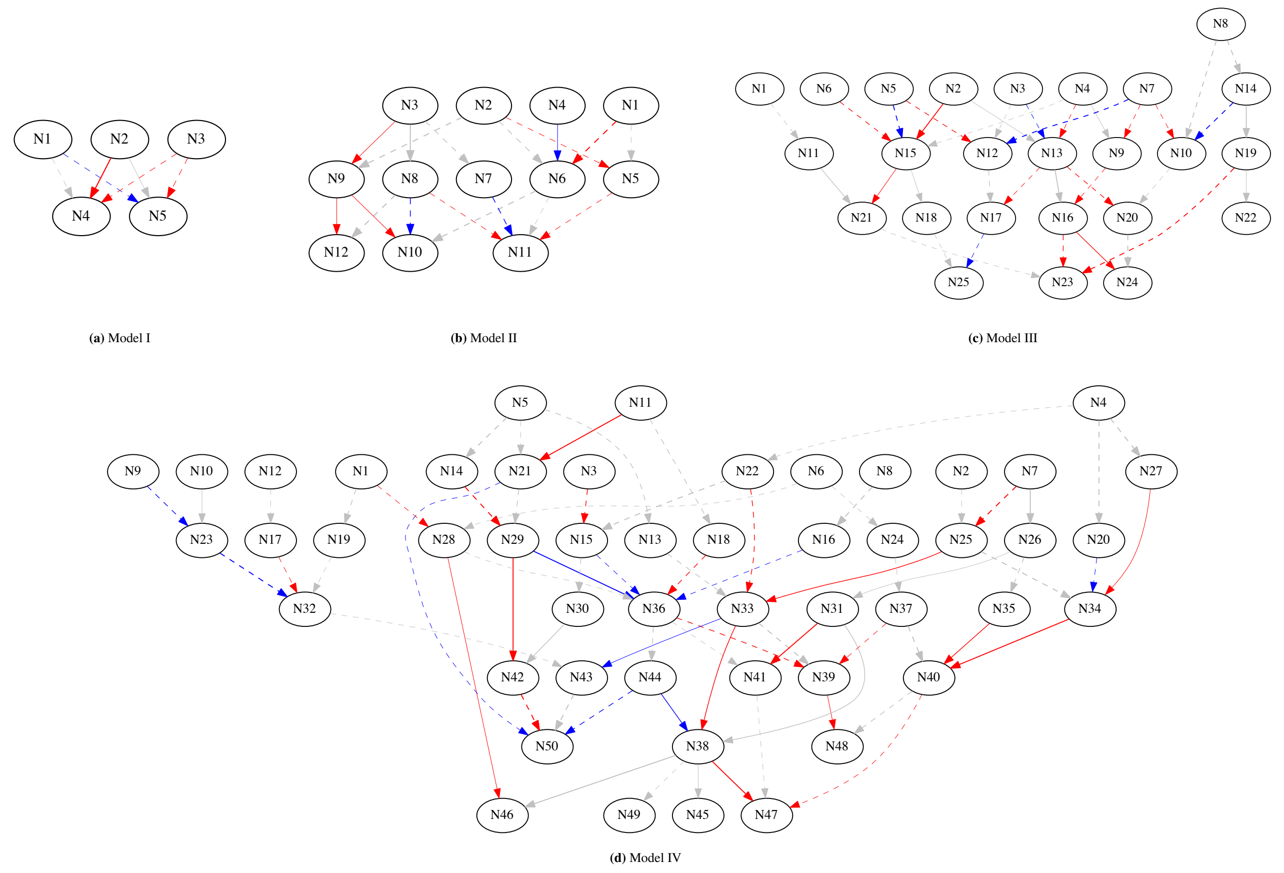

FIGURE 4 Models used in the experiments that were conducted to compare the solution time between the different solution approaches depicted in Figures 5 and 6. The color of the arrows denote de type of relation (univalent, bivalent or sweet point), dashed line denote nonlinear relations, while the thickness indicates the value of ρ_{ij}.

In order to test the influence of the initial points in finding the optimum, four different random initial points were considered for each problem and solution method. Feasible starting points are easy to find, since the only constraints are those imposed by the FCM model. This way, feasible points can be found choosing appropriate values for the input nodes (nodes without any incoming arcs) and computing the values of the rest of nodes of the model.

In turn, the objective functions used in the tests were composed of l_1 deviation penalization and linear terms:

$$\text{minimize } J = \sum_{i=1}^{N-2} C_i\, y_i + \sum_{i=N-1}^{N} C_i |y_i - r_i|, \tag{18}$$

where C_i represents the cost assigned to each term, and r_i symbolizes the target value for the variable. This type of functions capture well the typical requirements of penalizing the control effort while allowing the values of a variable to match a specific reference.

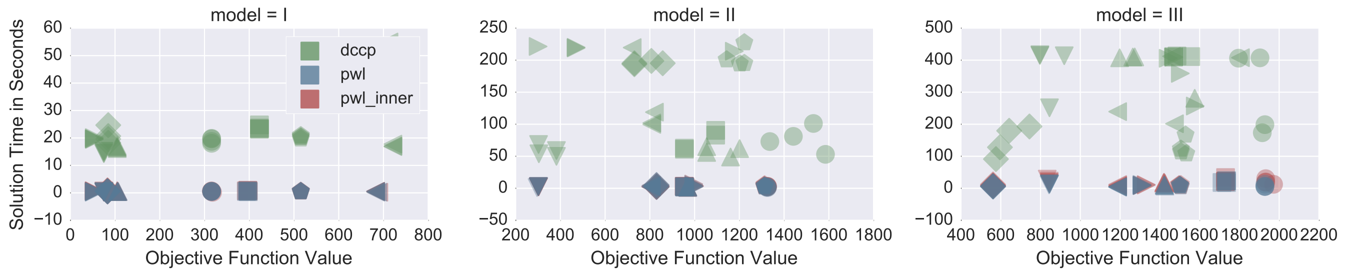

FIGURE 5 Solution time (y-axis) and optimal objective values (x-axis) obtained for models I, II and III (see Figure 4) for the different solution methods. Horizontal spread of markers denotes variability in the solutions provided by the solution method. Vertical spread designate variability in the solution time. Different markers stand for different relations between nodes for the same model topology. Please, note the different scales both in x and y axis for the different models.

The computational experiments were carried out in a mid-2012 MacBook Pro equipped with an Intel Core i7 and 8 GB RAM. The optimization problems were defined using CVXPY[27] and the DCCP extension for the convex-concave approach[20]. The solver used was Gurobi Optimizer version 6.5.1[28] for the mixed-integer problems and ECOS[29] for the convex subproblems.

Figure 5 shows the objective values obtained (x-axis) and the solution time required (y-axis) for the first three models and all the solution methods. The fourth model is not shown as the solution time required by the *dccp* method was prohibitely high and the experiments were aborted before completion. The first important conclusion that can be drawn from the inspection of this Figure, is the clear superiority of the separable programming methods over the convex-concave approach (marked as *dccp* in the Figure). For all three models, the solution time is much higher for the *dccp* method. Although the *dccp* method solves very fast the convexified subproblems, a large number of these subproblems needs to be solved, thus requiring a large time to solve the problem. Furthermore, for models II and III, the horizontal spread of the markers denotes the high variability in the objective values provided by the *dccp* method, as opposed to the concentration of the values provided by the separable programming methods. The theoretical lack of guarantee of global optimality of the solution found by the *dccp* method is in fact practically noticeable in this spread of values.

Figure 6 offers a closer look at the solutions obtained using the separable programming methods, including also the results obtained for model IV. As depicted in this Figure, the points corresponding to the same experiment tend to be fairly close, both in terms of objective value and solution time. This means that the initial points do not have a large influence on the optimal solution, nor on the computation time required to find it. This is very desiderable for optimization problems where obtaining a global optimum is not guaranteed, as a common practice in that scenario is to solve the problem several times using different starting points. This concentration of values alleviates the need to use multiple starting points, as the separable programming methods are shown to perform consistently for this problem reformulation.

It can be observed that the outer linearization method, denoted as *pwl* in the plots, tends to provide lower solution times than the inner approach, with this difference increasing with the size of the model. The average solution time for the different

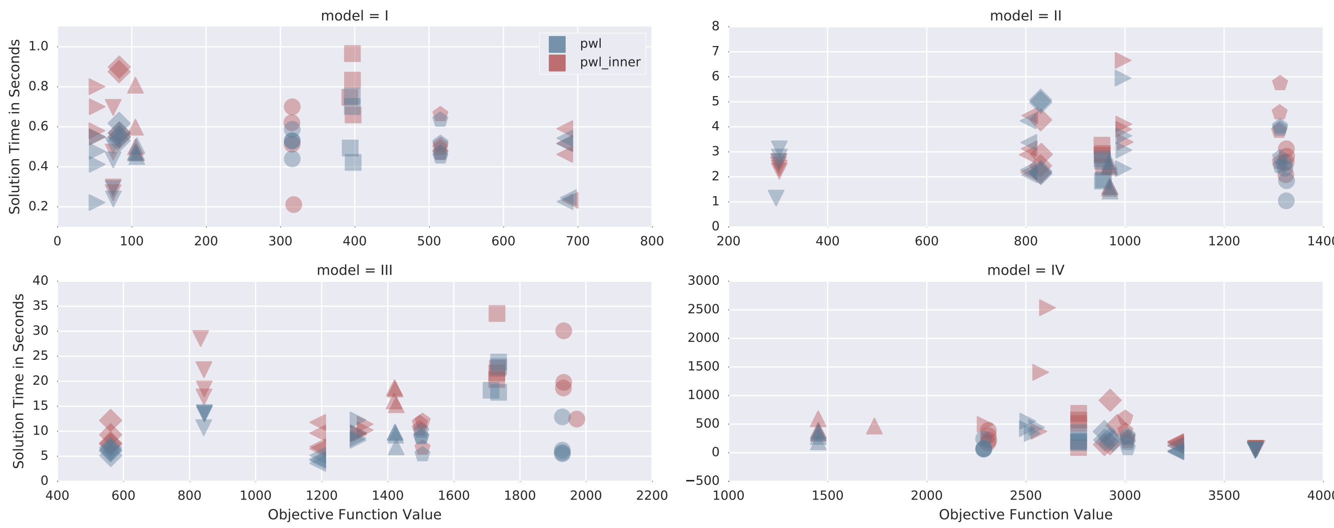

FIGURE 6 Solution time (y-axis) and optimal objective values (x-axis) obtained for models I, II, III and IV (see Figure 4) for the piecewise-linear methods. Horizontal spread of markers denotes variability in the solutions provided by the solution method. Vertical spread designate variability in the solution time. Different markers stand for different relations between nodes for the same model topology. Please, note the different scales both in x and y axis for the different models.

TABLE 1 Average solving time in seconds and standard deviation for the different models and solution methods.

	Method		
Model	dccp	pwl	pwl_inner
I	20.22 (6.93)	0.48 (0.13)	0.59 (0.19)
II	127.63 (69.30)	2.78 (1.14)	3.09 (1.12)
III	301.01 (122.13)	9.80 (5.15)	15.21 (7.08)
IV		205.31 (144.17)	427.05 (477.46)

models and solution methods can be found in Table 1, and shows the fast increase of the solution time as the model size grows, particularly between models III and IV. The order of magnitude of these computation times, however, renders them viable for their application in hierarchical control schemes, even for model IV, as set point update is typically performed at sample times that are larger than the values included in the table, and much larger than the sampling times used at the low-level control layers.

5 | CONCLUSIONS

This paper has introduced a reformulation of the nonlinear constraints provided by FCM and a comparison of two heuristic solution methods for the resulting problem. The reformulation introduced in Section 3 allowed to pose the problem as separable, which, in turn, allowed to use separable programming methods. The experiments have shown that the outer linearization method is the one that performed the best, providing reliability in finding the optimum from different starting points and reasonable solution times for moderate-sized models (25 nodes) and larger times, although still useful, for larger problems (50 nodes).

The reduction of the time required to solve an optimization problem provided by the proposed reformulation and solution method facilitates the use of FCM models in the upper layers of hierarchical control schemes. Also, the fact that only a Mixed-Integer Linear solver is needed for the problem, also expedites the practical implementation of the approach, be it with commercial solvers, such as Gurobi, or with open source ones, such as GLPK or lp_solve. This way, FCM becomes an interesting approach for the modeling of the relations between the high-layer variables in situations where data-driven approaches, based on access to online measurements of the involved variables, are not possible.

ACKNOWLEDGMENTS

This research was partially funded by Spanish Ministry of Science and Innovation under the project PID2019-110291RB-I00.

References

1. Tatjewski P. *Advanced Control of Industrial Processes: Structures and Algorithms*. London: Springer . 2010.
2. Qin SJ, Badgwell TA. A Survey of Industrial Model Predictive Control Technology. *Control Engineering Practice* 2003; 11(7): 733–764. doi: 10.1016/S0967-0661(02)00186-7
3. Ioannou I, Perrot N, Curt C, et al. The Fuzzy Symbolic Approach for the Control of Sensory Properties in Food Processes. In: Ruan PDD, Zeng PDX., eds. *Intelligent Sensory Evaluation*Springer Berlin Heidelberg. 2004 (pp. 175–195).
4. Perrot N, Ioannou I, Allais I, Curt C, Hossenlopp J, Trystram G. Fuzzy Concepts Applied to Food Product Quality Control: A Review. *Fuzzy Sets and Systems* 2006; 157(9): 1145–1154. doi: 10.1016/j.fss.2005.12.013
5. Cano Marchal P, Gómez Ortega J, Gámez García J. *Production Planning , Modeling and Control of Food Industry Processes*. Springer Nature . 2019
6. Kosko B. Fuzzy Cognitive Maps. *International Journal of Man-Machine Studies* 1986; 24: 65–74.
7. Stylios CD, Groumpos PP. Fuzzy Cognitive Maps: A Model for Intelligent Supervisory Control Systems. *Computers in Industry* 1999; 39(3): 229–238. doi: 10.1016/S0166-3615(98)00139-0
8. Papageorgiou E, Salmeron J. A Review of Fuzzy Cognitive Maps Research During the Last Decade. *IEEE Transactions on Fuzzy Systems* 2013; 21(1): 66–79. doi: 10.1109/TFUZZ.2012.2201727
9. Felix G, Nápoles G, Falcon R, Froelich W, Vanhoof K, Bello R. A review on methods and software for fuzzy cognitive maps. *Artificial Intelligence Review* 2019; 52(3): 1707–1737. doi: 10.1007/s10462-017-9575-1

10. Lu W, Feng G, Liu X, Pedrycz W, Zhang L, Yang J. Fast and Effective Learning for Fuzzy Cognitive Maps: A Method Based on Solving Constrained Convex Optimization Problems. *IEEE Transactions on Fuzzy Systems* 2020; 28(11): 2958–2971. doi: 10.1109/TFUZZ.2019.2946119

11. Salmeron JL, Mansouri T, Moghadam MRS, Mardani A. Learning Fuzzy Cognitive Maps with modified asexual reproduction optimisation algorithm. *Knowledge-Based Systems* 2019; 163: 723–735. doi: 10.1016/j.knosys.2018.09.034

12. Mls K, Cimler R, Vaščák J, Puheim M. Interactive evolutionary optimization of fuzzy cognitive maps. *Neurocomputing* 2017; 232(December 2016): 58–68. doi: 10.1016/j.neucom.2016.10.068

13. Cano Marchal P, García JG, Ortega JG. Application of Fuzzy Cognitive Maps and Run-to-Run Control to a Decision Support System for Global Set-Point Determination. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 2017; 47(8): 2256-2267. doi: 10.1109/TSMC.2016.2646762

14. Salehi HA, Tavassoli B. A gradient algorithm for solution of the optimal control problem for hybrid switching systems. *Optimal Control Applications and Methods* 2020; 41(6): 1854–1874. doi: 10.1002/oca.2673

15. Mall K, Grant MJ, Taheri E. Solving complex optimal control problems with nonlinear controls using trigonometric functions. *Optimal Control Applications and Methods* 2020(October): 1–13. doi: 10.1002/oca.2692

16. Keil RE, T.ăMiller A, Kumar M, Rao AV. Method for solving chance constrained optimal control problems using biased kernel density estimators. *Optimal Control Applications and Methods* 2021; 42(1): 330–354. doi: 10.1002/oca.2675

17. Cano Marchal P, Martínez Gila D, Gámez García J, Gómez Ortega J. Stochastic season-wide optimal production planning of virgin olive oil. *Journal of Process Control* 2018; 72: 64 - 73. doi: https://doi.org/10.1016/j.jprocont.2018.08.001

18. Cano Marchal P, Wagner C, García JG, Ortega JG. Modelling Uncertainty in Production Processes Using Non-Singleton Fuzzification and Fuzzy Cognitive Maps - a Virgin Olive Oil Case Study. In: 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). ; 2016: 1173–1180

19. Garzón Casado A, Cano Marchal P, Gómez Ortega J, Gámez García J. Visualization and Interpretation Tool for Expert Systems Based on Fuzzy Cognitive Maps. *IEEE Access* 2019; 7: 6140-6150. doi: 10.1109/ACCESS.2018.2887355

20. Shen X, Diamond S, Gu Y, Boyd S. Disciplined Convex-Concave Programming.. In: 2016 IEEE Conference on Decision and Control (CDC). ; 2016: 1009–1014.

21. Bradley H, Magnanti . *Applied Mathematical Programming*. Addison Wesley . 1977.

22. Lipp T, Boyd S. Variations and Extension of the Convex–concave Procedure. *Optimization and Engineering* 2016; 17(2): 263–287. doi: 10.1007/s11081-015-9294-x

23. Agrawal A, Boyd S. Disciplined quasiconvex programming. *Optimization Letters* 2020; 14(7): 1643–1657. doi: 10.1007/s11590-020-01561-8

24. Liberti L. *Reformulation and Convex Relaxation Techniques for Global Optimization*. PhD thesis. Imperial College, London; 2004.

25. Boyd SP, Vandenberghe L. *Convex Optimization*. Cambridge University Press . 2004.

26. Beale E, Forrest JJ. Global Optimization Using Special Ordered Sets. *Mathematical Programming* 1976; 10(1): 52–69.

27. Diamond S, Boyd S. CVXPY: A Python-Embedded Modeling Language for Convex Optimization. *Journal of Machine Learning Research* 2016; 17(83): 1–5.

28. Gurobi Optimization I. Gurobi Optimizer Reference Manual. 2015.

29. Domahidi A, Chu E, Boyd S. ECOS: An SOCP solver for embedded systems. In: European Control Conference (ECC). ; 2013: 3071-3076.

$\square$