Generating Conflict-Free Treatments for Patients with Comorbidity using ASP

Elie Merhej[1], Steven Schockaert[2], T. Greg McKelvey[4⋆], and Martine De Cock[1,3]

[1] Ghent University, Ghent, Belgium
{elie.merhej,martine.decock}@ugent.be
[2] Cardiff University, Cardiff, United Kingdom
schockaerts1@cardiff.ac.uk
[3] University of Washington Tacoma, Tacoma, USA
mdecock@u.washington.edu
[4] KenSci, Seattle, USA
Greg@KenSci.com

Abstract. Conflicts in recommended medical interventions regularly arise when multiple treatments are simultaneously needed for patients with comorbid diseases. An approach that can automatically repair such inconsistencies and generate conflict-free combined treatments is thus a valuable aid for clinicians. In this paper we propose an answer set programming based method that detects and repairs conflicts between treatments. The answer sets of the program directly correspond to proposed treatments, accounting for multiple possible solutions if they exist. We also include the possibility to take preferences based on drug-drug interactions into account while solving inconsistencies. We show in a case study that our method results in more preferred treatments.

1 Introduction

Clinical Practice Guidelines (CPGs) are documents created by experts in the medical field in order to help clinicians in treating certain diseases [8]. To make CPGs more accessible and easier to use, many frameworks have been developed that gather important information from a relevant CPG and transform it into a computer interpretable format [2,6,10]. The resulting representation is called a Computer Interpretable Guideline (CIG). These frameworks usually create task networks that represent possible treatments of the disease based on certain actions, decisions and tests. When a physician wants to treat a specific disease, they consult the corresponding task network and follow the presented procedure based on the test results that are available for the patient. A lot of research has been done to improve CIGs by including, for example, patient-specific information in the generated task networks [11,9], different kinds of treatments [1] and even rules about hospital and insurance policies in order to create the best

⋆ University of Washington Occupational and Environmental Medicine Fellow

possible personalized treatment for every patient. In addition, research has been done for the verification of the resulting medical guidelines [3].

It is important, however, to note that CPGs were originally designed to treat every disease separately. Recent studies have shown that the number of patients with comorbidity, i.e. diagnosed with multiple diseases simultaneously, keeps rising [5]. Combining CIGs for individual diseases regularly results in conflicts between the recommended treatments. These conflicts can be on different levels: drug-drug interactions, drug-disease interactions, etc. Often these conflicts can be solved by the expertise of the physician, especially when swapping a drug by another one could resolve the conflict. Given the increasing number of CIGs and the complexity of the presented treatments, a system that can automatically detect [13] and repair inconsistencies when combining multiple CIGs [4] would be a valuable aid for clinicians.

Building on recent work by Wilk et al. [12], implemented in Zhang et al. [14], we present in this paper an answer set programming (ASP) based approach for generating conflict-free treatments for comorbid patients. ASP is a declarative problem solving language, which allows one to describe a problem as a set of logical rules. ASP solvers are then used to find answer sets, i.e. the sets of facts that satisfy all the encoded rules. In our case, the ASP rules represent the method to detect and repair conflicts in candidate treatments, while the answer sets correspond to valid combined treatments. Removing inconsistencies in treatments involves applying mitigation operators (MOs) [12] which by themselves can introduce undesirable drug-drug interactions. Therefore, we define preferences among the answer sets by assigning a penalty based on the drug-drug interactions they contain. This penalty is equal to the sum of individual penalties introduced by every drug-drug interaction found, which depends on the severity of the corresponding interaction. Treatment penalties induce a ranking from the most preferred valid treatment (i.e. with the smallest penalty) to the least preferred (i.e. with the highest penalty). The main differences between our approach and the work by Zhang [14] are:

- All the answer sets that we generate are valid solution treatments.
- We apply one MO at a time, instead of all simultaneously, until a point of contention is eliminated.
- We introduce preferences among solution treatments based on drug-drug interactions, to select the most desirable treatment among the given candidates.

The remainder of the paper is structured as follows: after recalling some preliminaries about ASP in Section 2, in Section 3 we present a method to resolve conflicts that arise when applying multiple CPGs on a patient with comorbid diseases, and rank solution treatments based on drug-drug interactions. In Section 4, we show the advantages of our approach by introducing a case study that involves the task networks from the CPGs for duodenal ulcer (DU) and transient ischemic attack (TIA). Finally, we conclude in Section 5.

2 Answer Set Programming

Answer Set Programming (ASP) is a declarative problem solving language [7], which allows one to describe a problem as a set of rules of the form $h \leftarrow a_1, \ldots, a_j, not\ b_{j+1}, \ldots, not\ b_k$. ASP solvers can then find the answer sets (see below) which correspond to the solutions of the encoded problem. The head and the body of an ASP rule r are respectively defined as $head(r) = h$ and $body(r) = \{a_1, \ldots, a_j, not\ b_{j+1}, \ldots, not\ b_k\}$. The "," in the body of r represents a conjunction. If $body(r) = \emptyset$, then r is called a *fact*. If $head(r) = \emptyset$, then r is called a constraint. Constraints act as filters on the possible answer sets. The keyword *not* represents *negation-as-failure* in ASP, where *not a* intuitively holds whenever we cannot derive that a holds. Let $body^+(r) = \{a_1, \ldots, a_j\}$ and $body^-(r) = \{b_{j+1}, \ldots, b_k\}$. A set of atoms X is closed under Π if for any rule $r \in \Pi$, $head(r) \in X$ whenever $body^+(r) \subseteq X$. The smallest set of atoms closed under Π is denoted by $Cn(\Pi)$. The reduct Π^X of Π relative to X is defined by $\Pi^X = \{head(r) \leftarrow body^+(r) \mid r \in \Pi \text{ and } body^-(r) \cap X = \emptyset\}$. A set X of atoms is called an *answer set* (i.e. stable model) of Π if $Cn(\Pi^X) = X$. For example, let Π be the answer set program formed by the rule $c \leftarrow not\ b$, the rule $b \leftarrow not\ c$ and the fact a. This program has two answer sets $\{a, c\}$ and $\{a, b\}$.

3 Finding Preferred Conflict-Free Treatments

As in [14], we propose an ASP implementation of the theoretical method proposed in [12] to resolve conflicts that arise in the concurrent application of CPGs on a patient with multiple diseases. These CPGs are represented by task networks that contain their relevant information. We formulate our encoding such that every answer set corresponds directly to a solution treatment, which is different from the encoding proposed in [14]. The facts of our ASP program encode task networks (action nodes, decision nodes, edges and labels), patient information, MOs, and a given set of points of contention that can occur. To find solution treatments, first we pick a candidate treatment from the task network of every disease (rules 1-2), which is also compliant with the encoded patient information (rules 3-5).[5]

$$candidateEdge(Ag, X, Y) \leftarrow edge(Ag, X, Y),\ not\ nCandidateEdge(Ag, X, Y). \quad (1)$$

$$nCandidateEdge(Ag, X, Y) \leftarrow edge(Ag, X, Y),\ not\ candidateEdge(Ag, X, Y). \quad (2)$$

$$nodeInTreat(Ag, X) \leftarrow candidateEdge(Ag, X, Y). \quad (3)$$

$$nodeInTreat(Ag, Y) \leftarrow candidateEdge(Ag, X, Y). \quad (4)$$

$$\leftarrow dNode(Ag, X),\ nodeInTreat(Ag, X),\ patientInfo(X, L), \\ label(Ag, X, Y, L),\ not\ candidateEdge(Ag, X, Y). \quad (5)$$

Then, we detect active points of contention in every pair of candidate treatments by checking whether every action in the point of contention is in the selected candidate treatment. To eliminate every detected point of contention, we apply

[5] For the full code, see http://www.cwi.ugent.be/ComorbidityConflictSolver.html

an applicable MO (rules 6-7). A solution treatment consists then of a combination of candidate treatments modified by the applied MO (rules 8-10) that does not contain active points of contention (rules 11-14).

$$applyMO(PocID, MoID) \leftarrow activePOC(PocID),\ applicableMO(PocID, MoID),\ not\ napplyMO(PocID, MoID). \quad (6)$$

$$napplyMO(PocID, MoID) \leftarrow activePOC(PocID),\ applicableMO(PocID, MoID),\ not\ applyMO(PocID, MoID). \quad (7)$$

$$solutionTreat(TD, A) \leftarrow activeAction(TD, A),\ applyMO(PocID, MoID),\ moTD(MoID, TD),\ not\ moToBeRemoved(MoID, A). \quad (8)$$

$$solutionTreat(BD, A) \leftarrow applyMO(PocID, MoID),\ moBD(MoID, BD),\ moRHS(MoID, pos(A)). \quad (9)$$

$$solutionTreat(BD, A) \leftarrow activeAction(BD, A),\ applyMO(PocID, MoID),\ moBD(MoID, BD),\ not\ occursIn(A, MoID). \quad (10)$$

$$solutionAction(A) \leftarrow solutionTreat(D, A). \quad (11)$$

$$ignorePOC(PocID) \leftarrow not\ solutionAction(A),\ pocAction(PocID, A). \quad (12)$$

$$pocFound(PocID) \leftarrow poc(PocID),\ not\ ignorePOC(PocID). \quad (13)$$

$$\leftarrow pocFound(PocID). \quad (14)$$

Different from [14], where all applicable MOs are applied simultaneously, we apply only one MO at a time (rules 15-18). Applying multiple MOs to resolve the same point of contention has a high chance of introducing multiple drugs that have the same purpose in the same treatment. This is not advised for multiple reasons. First, it creates treatments that contain a higher dosage of a substance than originally intended. Second, the number of side effects that a patient may develop increases when the number of prescribed drugs increases, and third, avoidable drug-drug interactions may be introduced.

$$appliedMOs(PocID, X) \leftarrow X = \#count\{applyMO(PocID, MoID)\},\ activePOC(PocID). \quad (15)$$

$$errorApplyingMo(PocID) \leftarrow appliedMOs(PocID, X),\ X < 1. \quad (16)$$

$$errorApplyingMo(PocID) \leftarrow appliedMOs(PocID, X),\ X > 1. \quad (17)$$

$$\leftarrow errorApplyingMo(PocID). \quad (18)$$

Among all solution treatments found by the rules above, some might be preferred over others depending on the severity of drug-drug interactions that are introduced to the treatment after applying the corresponding MO. These interactions may persist in proposed valid solution treatments due to different factors. On one hand, some drug-drug interactions may only be partially described, i.e. only the cause of the interaction is known, but the way to mitigate it is still unknown. Hence, these interactions cannot be encoded in mitigation operators. We therefore propose to use them to penalize the obtained solution in case they are present. On the other hand, with the continuous improvements in clinical research, previously reliable MOs may become outdated, thus containing incomplete or inaccurate information. Applying these MOs to resolve conflicts when combining new treatments may then introduce drug-drug interactions that were previously undetected.

To induce a ranking among solution treatments, we encode in our program facts of the form "$drug(X)$" that read "X is a drug" and facts of the form

"$interaction(X, Y, C)$" that read "drug X has an interaction level C with drug Y". These drug-drug interactions are included for every pair of drugs present in the treatments. Their respective interaction levels can be found in the medical literature (see Section 4). For every treatment we then assign a penalty that is equal to the sum of all the levels of interactions between the drugs in that treatment (rules 19-21). The treatment that minimizes this penalty is considered the best (rule 22). Note that this ASP encoding can output all possible treatments with their respective optimization value when specified by the ASP solver, and not only the best one.[6] This allows the ranking of all the treatments from most to least preferred.

$$solutionDrug(X) \leftarrow solutionAction(X),\ drug(X). \tag{19}$$
$$solutionInteraction(X, Y, C) \leftarrow interaction(X, Y, C),\ solutionDrug(X),\ solutionDrug(Y). \tag{20}$$
$$interactionsPenalty(P) \leftarrow P = \#sum[solutionInteraction(X, Y, C) = C]. \tag{21}$$
$$\#minimize[interactionsPenalty(P) = P]. \tag{22}$$

4 Case Study

We extend a use case from [14] to show the advantages of our approach. The example involves the task networks from the CPGs for duodenal ulcer (DU) and transient ischemic attack (TIA), shown in Fig. 1 [12], and a patient who is di-

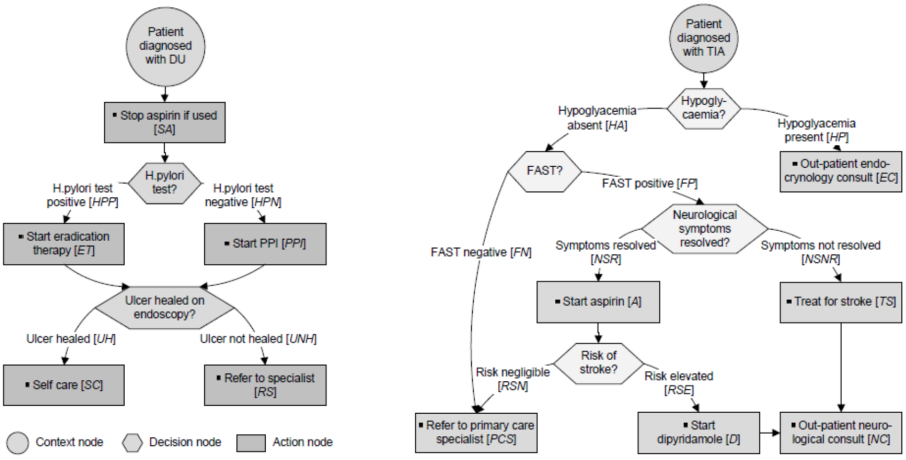

Fig. 1. Task networks for treating the DU disease (left) and the TIA disease (right).

[6] See http://www.cwi.ugent.be/ComorbidityConflictSolver.html

agnosed with both diseases. The following patient information is used: H.pylori test negative, ulcer healed, hypoglyacemia absent, FAST test positive, and neurological symptoms resolved. No data is included concerning the risk of stroke. Based on this information, one candidate treatment is extracted from the task network of DU: $CT^1_{du} = \{SA, PPI, SC\}$ and two candidate treatments from the task network of TIA: $CT^1_{tia} = \{A, PCS\}$ and $CT^2_{tia} = \{A, D\}$. We use the following four MOs:

1. MO_1: $\{tia, du, \{A, SA\}, \{pos(A)^7, neg(D)\}, \{neg(A), pos(Cl)\}, SA\}$
2. MO_2: $\{tia, du, \{A, SA\}, \{pos(A), pos(D)\}, \{pos(A), pos(D), pos(PPI)\}, SA\}$
3. MO_3: $\{tia, du, \{A, SA\}, \{pos(A), neg(D)\}, \{pos(A), pos(Cy^8)\}, SA\}$
4. MO_4: $\{tia, du, \{A, SA\}, \{pos(A), neg(D)\}, \{pos(A), pos(Fl^9)\}, SA\}$

For this scenario, our ASP rules 1-18 from Section 3 generate four answer sets: ST_4, ST_6, ST_7 and ST_9 shown in Table 1. One can verify that ST_4, ST_6 and ST_7 result from applying MO_3, MO_4 and MO_1 respectively to the combination of the treatments CT^1_{du} and CT^1_{tia}. Each of these MOs correctly removes the conflict introduced by the actions "A" and "SA". Similarly, the solution treatment ST_9 arises from applying MO_2 to the combination of treatments CT^1_{du} and CT^2_{tia}.

Name	**Answer Set**	**MOs Applied**	**Penalty**	**Zhang's Approach**	**Preference Based Approach**
ST_1	{*PPI, SC, A, PCS, Cl, Cy, Fl*}	MO_1, MO_3, MO_4	7	Y	N
ST_2	{*PPI, SC, A, PCS, Cy, Fl*}	MO_3, MO_4	4	Y	N
ST_3	{*PPI, SC, A, PCS, Cl, Cy*}	MO_1, MO_3	5	Y	N
ST_4	{*PPI, SC, A, PCS, Cy*}	MO_3	2	Y	Y (2)
ST_5	{*PPI, SC, A, PCS, Cl, Fl*}	MO_1, MO_4	6	Y	N
ST_6	{*PPI, SC, A, PCS, Fl*}	MO_4	3	Y	Y (3)
ST_7	{*PPI, SC, PCS, Cl*}	MO_1	3	Y	Y (3)
ST_8	{} - Inv.	Inv.	Inv.	Y	N
ST_9	{*PPI, SC, A, D, NC*}	MO_2	1	Y	Y (1)
ST_{10}	{} - Inv.	Inv.	Inv.	Y	N

Table 1. Solution treatments found by Zhang's approach, and by our preference based approach. "Inv." indicates an invalid treatment. "Y" indicates that a treatment is given as a solution by the approach. "N" indicates that a treatment is not given as a solution by the approach. "Y(P)" indicates that a treatment is given as a solution by the preference based approach with the rank P.

Next, rules 19-22 can be used to induce a preference ranking over the valid solution treatments, based on the interaction levels of the drugs in the proposed treatments. We consider four levels of interactions: major, moderate, minor and

[7] The keywords *pos(X)* and *neg(X)* refer to an action X being present and absent from a treatment respectively.

[8] Cy: Cyanocobalamin

[9] Fl: Flibanserin

no interaction, represented correspondingly in our ASP program by 3, 2, 1 and 0.[10] Now, running the same scenario in our ASP program identifies a preferred solution treatment: ST_9. In fact, ST_7 contains 1 major drug-drug interaction between Cl and PPI (penalty=3), ST_9 contains 1 minor drug-drug interaction between A and PPI (penalty=1), ST_4 contains 2 minor drug-drug interactions between A and PPI, and between Cy and PPI (penalty=1+1=2), and ST_6 contains 1 minor drug-drug interaction between A and PPI, and 1 moderate drug-drug interaction between Fl and PPI (penalty=1+2=3). With ST_9 having the lowest penalty, it is indeed the best possible treatment (rank=1).

Running the same scenario using Zhang's program [14] gives 10 answer sets, shown in Table 1. In addition to the solutions found by our preference based approach, Zhang's program gives 2 invalid solution treatments where no MOs are applied: ST_8 and ST_{10}, and 4 answer sets where multiple MOs are applied simultaneously: ST_1, ST_2, ST_3 and ST_5. These last 4 solution treatments have penalties equal to 7, 4, 5 and 6 respectively, which are higher than penalties of solution treatments found by applying one MO at a time. This shows that applying multiple MOs concurrently may introduce avoidable and potentially more dangerous drug-drug interactions.

5 Conclusion

The number of patients diagnosed with multiple diseases is rising. In this preliminary paper, we presented an extension of Zhang and Zhang's ASP encoding for the problem of generating conflict-free treatments for patients with comorbidity [14]. A noteworthy difference between our approach and the work in [14] is that all answer sets of our ASP program directly correspond to solution treatments, which makes it arguably easier to use by physicians. In addition, our ASP program adheres closer to [12] in applying one MO at a time instead of all simultaneously, in the case where multiple applicable MOs are available. This distinction is clinically important because applying multiple MOs in parallel has a high chance of introducing multiple drugs that have the same purpose in the same treatment. Furthermore, we refined our ASP program with a ranking mechanism based on the severity of drug-drug interactions in solution treatments, thereby providing a technique to identify preferred treatments. An interesting direction for future research involves expanding the current ASP encoding with multiple ways of defining preferences between treatments, such as different types of interactions (drug-disease, drug-food, etc.) to complement the drug-drug interactions that we already included. We also plan to work with multiple medical experts in order to create a better encoding of the candidate treatments specified in clinical guidelines that accounts for the time variable, and more specific action nodes.

[10] Drug-drug interactions obtained from the "Interactions Checker" at www.drugs.com

References

1. Barr, J., Fraser, G.L., Puntillo, K., Ely, E.W., Gélinas, C., Dasta, J.F., Davidson, J.E., Devlin, J.W., Kress, J.P., Joffe, A.M., et al.: Clinical practice guidelines for the management of pain, agitation, and delirium in adult patients in the intensive care unit. Critical care medicine 41(1), 263–306 (2013)
2. De Clercq, P.A., Blom, J.A., Korsten, H.H., Hasman, A.: Approaches for creating computer-interpretable guidelines that facilitate decision support. Artificial intelligence in medicine 31(1), 1–27 (2004)
3. Hommersom, A., Groot, P., Lucas, P.J., Balser, M., Schmitt, J.: Verification of medical guidelines using background knowledge in task networks. IEEE Transactions on Knowledge and Data Engineering 19(6), 832–846 (2007)
4. Jafarpour, B., Abidi, S.S.R.: Merging disease-specific clinical guidelines to handle comorbidities in a clinical decision support setting. In: Conference on Artificial Intelligence in Medicine in Europe. pp. 28–32. Springer (2013)
5. Jakovljevic, M., Ostojic, L.: Comorbidity and multimorbidity in medicine today: challenges and opportunities for bringing separated branches of medicine closer to each other. Psychiatr Danub 25(Suppl 1), 18–28 (2013)
6. Latoszek-Berendsen, A., Tange, H., Van Den Herik, H., Hasman, A., et al.: From clinical practice guidelines to computer-interpretable guidelines. Methods of information in medicine 49(6), 550–570 (2010)
7. Lifschitz, V.: What is answer set programming?. In: AAAI. vol. 8, pp. 1594–1597 (2008)
8. Panel, D.: Clinical practice guidelines. Vol I. Washington, DC: Agency for Health Care Policy and Research (1993)
9. Spiotta, M., Bottrighi, A., Giordano, L., Dupré, D.T.: Conformance analysis of the execution of clinical guidelines with basic medical knowledge and clinical terminology. In: Workshop on Knowledge Representation for Health-Care Data, Processes and Guidelines. pp. 62–77. Springer (2014)
10. Ten Teije, A., Miksch, S., Lucas, P.: Computer-based medical guidelines and protocols: a primer and current trends, vol. 139. Ios Press (2008)
11. Tu, S.W., Campbell, J.R., Glasgow, J., Nyman, M.A., McClure, R., McClay, J., Parker, C., Hrabak, K.M., Berg, D., Weida, T., et al.: The sage guideline model: achievements and overview. Journal of the American Medical Informatics Association 14(5), 589–598 (2007)
12. Wilk, S., Michalowski, W., Michalowski, M., Farion, K., Hing, M.M., Mohapatra, S.: Mitigation of adverse interactions in pairs of clinical practice guidelines using constraint logic programming. Journal of biomedical informatics 46(2), 341–353 (2013)
13. Zamborlini, V., Hoekstra, R., Da Silveira, M., Pruski, C., ten Teije, A., van Harmelen, F.: Inferring recommendation interactions in clinical guidelines. Semantic Web 7(4), 421–446 (2016)
14. Zhang, Y., Zhang, Z.: Preliminary result on finding treatments for patients with comorbidity. In: Knowledge Representation for Health Care, pp. 14–28. Springer (2014)