Individual Project Report

Shaimaa Ibrahim
Ira A.Fulton Schools of Engineering
Arizona State University
sibrah10@asu.edu

Abstract

A company assigns referees to different regions to validate insurance claims. The purpose of this project is to deploy ASP (Answer Solution Programming) to assign referees to insurance cases with hard and weak constraints imposed. Although more than one referee may be eligible for a case, the given optimization criteria gives the best possible solution.

Project Summary

ASP is a type of declarative programming that is based on representing the knowledge rather than coming up with an algorithm to find the solution. This means that the skills of an ASP programmer lie in how well they can write a code to represent the given knowledge for the ASP solver to display the right Information. The findings of this report are based on the completion of CSE579 project 1, which is adopted from challenge hosted by a European universities. Out of the 2 choices given, the *Insurance Referee Assignment Problem* was selected.

The problem statement of this project is assigning a given case to the fittest referee base on predefined rules known as hard and weak constraints. Cases are insurance claims and referees are potential candidates. Imposed hard constraints do not guarantee the finding one stable mode as more than one referee may satisfy all the hard constraints. Therefore, weak constraints are further introduced to attain the completeness of the program.

Although there are many other ASP systems, Clingo was used to solve the given problem. It is possible to come up with an optimal solution by solely using logic and a machine to compute all the calculation. The catch is that human methods are time consuming and prone to error. ASP offers speed and accuracy.The basic knowledge of the basics of ASP and Clingo gained from CSE579 lectures and previous assignments was sufficinet to complete this project; no additional programming tools were required. Cross-validation of the optimization scores for the stable models was performed manually.

Besides the hard and weak constraints provided, a set of parameters are set for each referee and each case. For every assignment problem, an instance is provided, where it consists of at least one case and two or more referees with pre-defined input arguments. Such parameters are linked together by atoms. The atom for the referee consists of 5 parameters:

- Referee ID (*Rid*): it is a unique identification for a referee,
- Referee type (*Type*): can be either external or internal.
- Maximum workload (*maxworkload*): the maximum allowed working minutes for a referee per day.
- Previous workload (*prevworkload*): the cumulative working minutes of previous cases.
- Previous payment (*prevpayment*): only applicable to external referees as internal referees are paid a fixed salary. So, by default, this parameter is assigned to 0 form internal referees.

Another atom with 6 parameters is used to define a case:

- Case ID (*Cid*): it is a unique identification for a case.
- Case type (*Caset*): Alphabetical characters that represent a case type.
- Case Effort (*Effort*): the number of daily working minutes to be added to the referee's existing workload if they are assigned the case.
- Case Damage (*Damage*): the amount of damage in Euros of the insurance claim case.
- Case Postal Code (*Postc*): A postal code that indications the geographical area of the case.
- Case Payment (*Payment*): The payment to be a made if the case is assigned to an external referee. Should not be considered for internal referees.

Three following 3 atoms that link the referees and the cases are typically part of each intance:

- *PrefType(Rid,Caset,Pref*, where *pref* is an integer that links a referee's preference to a case type.
- *PrefRegion(Rid,Postc,Pref*, where *pref* is an integer that links a referee's preference to a a region. The region of a case can be known form the *postc*.

- *externalmaxdamage(d)*, which d is a variable that specifies the damage threshold for a given case. External referees can only handle cases below the damage threshold.

The definition of the constraints in the coming section will provide a solid understanding of the atoms defined in the problems' instances.

Approach to finding the Optimal Solution

The basic idea of ASP is finding every possible combination or solution of a given instance. In this assignment problem, every case must be assigned to only one referee, while a referee may be assigned more than one case. In ASP language. This can be express as follows. The assignment relationship is represented by atoms that link the parameters together.

$$1 assign(Cid, Rid) : referee(Rid,_{,\,,,})1 :- case(Cid,_{,\,,,,}). \tag{1}$$

In the absence of constraints, which are merely rules of what is allowed and disallowed, many stable models will be generated. This defies the purpose of the assignment. Hard constraints work by eliminating ineligible referees, however in real-life, it is usually not all-or-nothing; the solution is about finding the fittest candidate. The main hard constraints written in ASP to be run on Clingo are listed below. However, it is important to note that his not the full functioning code.

Writng the Hard Constraints

- *Hard constraint 1*
  The snippet of code below shows that if the total workload of one case or a number of cases exceeds the maximum workload of a referee, then such case(s) combination is not allowed.

$$:-referee(Rid,_{,} Max,_{,\,}), sumC, A, B : agg(A, B, C) > Max, B = Rid, case(Cid,_{,\,,,,}). \tag{2}$$

- *Hard constraint 2*
  A referee with a preference input agrument for prefRegion of 0 cannot be assigned this case with matching PostalCode(Postc).

$$:-assign(Cid, Rid), referee(Rid,_{,\,,,}), prefRegion(Rid, Postc, P), case(Cid,_{,\,,}Postc, 0) \tag{3}$$

item *Hard constraint 2*
A referee with a preference input agrument for prefRegion of 0 cannot be assigned this case with matching PostalCode(Postc).

$$:-assign(Cid, Rid), referee(Rid,_{,\,,,}), prefRegion(Rid, Postc, P), case(Cid,_{,\,,}Postc, 0) \tag{4}$$

item *Hard constraint 3*
A referee with a preference input agrument for prefType of 0 cannot be assigned this case with matching Case type(Caset).

$$:-assign(Cid, Rid), referee(Rid,_{,\,,,}), prefType(Rid, Caset, P), case(Cid, Caset,_{,\,,}0). \tag{5}$$

item *Hard constraint 4*
Cases with an amount of damage that exceeds a certain threshold can only be internal referees.

$$:-assign(Cid, Rid), referee(Rid, e,_{,\,,}), case(Cid,_{,\,}, D1,_{,\,}), D1 > d : externalMaxDamage(d). \tag{6}$$

Writing the Weak Constraints

Weak constraints are also rules, except that they can be violated at a cost known as a penalty. This is the key to choosing the best stable model as usually the solution constitutes more than one stable model. Some assumptions were associated with each weak constraint as follows:

- *Weak Constraints 1*
  In case no external referee is allowed, the total payment (**CA**) must be 0, where no penalty is incurred as constraint 1 is satisfiable. If the hard constraints allow external referees, then the combination of external referees that gives the minimum total payment will result in the lowest penalty score. It is preferable that the penalty constraint is stored on a different level to help debug the code. For the first constraint the weight is 16 and the level is 0, which means that for all solutions the penalty at level is 0 is only related to the total payment of external referees.

$$:\ assign(C1, R1), referee(R1, T,_{,\,,}), T = e, case(C1,_{,\,,,}S1).[16 * S1@0] \tag{7}$$

- *Weak constraint 2*
  In order balance the overall payment of external referees, a penalty equivalent to the deviation from the overall payment average (**CB**)is imposed. If an external referee is excluded due to violating a hard constraint, he/she will not be included in the average calculation.

$$:\ referee(R1, e,_{,\,,}), orid(R1, S)avgp(S1). [7 * |S - S1|@1] \tag{8}$$

- *Weak constraint 3*
  The workload of all eligible referees should be balanced, a penalty equivalent to the deviation from the overall workload average (**CC**)is imposed. If an external referee is excluded due to violating a hard constraint, he/she will not be included in the average calculation. It is also important to factor in every possible case-referee combination when calculating the average.

$$:\ assign(C1, R1), referee(R1,_{,\,,,}), wrid(R1, S), avgw(S1), not\ agg3(C1, R1), agg2(C1, R1,_{)}.[9 * |S - S1|@2] \tag{9}$$

- *Weak constraint 4*
  Referees that do not violate *constraint 3* but have a case type preference less than 3 should also face a penalty (**CD**). The penalty is calculated by subtracting the referee's preference from 3; the higher the preference the lower the penalty. If the referee's case type preference is not specified in the problem, a preference of 0 is assigned to the referee and they may not be assigned the case.

$$: \ assign(Cid, R1), prefType(R1, Caset, P), case(Cid, Caset,_{,\,,,)}), not\ agg3(Cid, R1), agg(Cid, R1,_{)}).[34 * (3 - P)@4] \tag{10}$$

- *Weak constraint 5*
  Similar to *Weak Constraint 4*, referees that do not violate *constraint 2* but have a case region preference less than 3 should also face a penalty (**CE**). The penalty is calculated by subtracting the referee's preference from 3; the higher the preference the lower the penalty. If the referee's case region preference is not specified in the problem, a preference of 0 is assigned to the referee and they may not be assigned the case.

$$: \ assign(Cid, R1), prefRegion(R1, Postc, P), case(Cid,_{,\,,,}Postc_{)}), not\ agg3(Cid, R1), agg(Cid, R1,_{)}).[34 * (3 - P)@4] \tag{11}$$

Main Results and Analysis

Recall that every instance is a problem by itself. However, each problem only tests the rigidity of only a few hard and soft constraints. This means that to truly confirm that the code with working, it must pass all 10 test cases.

Example 1

Through the sole use of mental computation, problem one may be solved. To begin with, all only internal referee 5 is eligible to be assigned the given case (case 4) as the damage threshold exceeds the case's damage, which would violate *hard constraint 4*if referee 6 was assigned the case. Referee 4 cannot be assigned the case either as he/she has 0 preference for cases of type c, which is the type of case 4. Assigning a referee to case type with 0 preference violates *hard constraint 2*.Therefore, by elimination case 4 can only be assigned to referee number 5. The optimization score displayed by Clingo aligns with the manual computation done as per the optimizaton criteria:
No external referee cost is incurred as case 4 cannot be assigned to an external referee. Similarly, the average payment for external referees should be 0. The average overall workload is the same as the workload of referee 4 as he/she is the only eligible referee in this example. Given that referee 5 does not have a preference of 3 for both the case type and region, a penalty must be calculated accordingly. The calculation yields.

$$\begin{aligned} &C_A, C_B, C_C = 0 \\ &C_D, C_E = 3 - 2 \\ &C = c = 0 + 0 + 0 + 34 * 1 + 34 * 1 = 68 \end{aligned} \tag{12}$$

Example 2

Similar to Example 1, this problem may be solved by elimination. Assigning referees 9 or 10 to case 5 violates *Hard constraint 2*. Referee 7 incurs a penalty for having a region preference less than 3 as per *Hard constraint 2*. The manually computed solution also aligns with Clingo's displayed output.

$$\begin{aligned} &C_A, C_B, C_C = 0 \\ &C_D, C_E = 3 - 1 \\ &C = c = 0 + 0 + 0 + 0 + 34 * 2 = 68 \end{aligned} \tag{13}$$

Example 3

This example further tests the rigidity of the proposed solution as it cannot be solved by elimination, where relying of the hard constraints outputs 2 stable models. Only referee 10 is eliminated as the workload of case 6 exceeds his/her maximum daily working hours. Such condition violates **Hard constraint 1**. It is important to eliminate the overall payment and overall workload of referee 10 from the average overall payment and average overall workload to get the right optimum solution. The following is the optimum score, which belongs to referee 11.

$$\begin{aligned} &C_A = 80, C_B = |380 - 1630|, C_C = |210 - 275| \\ &C_D = 3 - 2, C_E = 3 - 2 \\ &C = c = 80 * 16 + 1250 * 7 + 65 * 9 \\ &\ + 34 * 1 + 34 * 1 = 10,683 \end{aligned} \tag{14}$$

Example 4

In this example, referee 15 must be excluded since the damage of case 7 exceeds the maximum external damage threshold. Therefore, assigning the case to an external referee violates *Hard constraint 4*. Referee 14 has a lower penalty score- as shown below- than referee 13; hence, he/she was assigned the case.

$$\begin{aligned} &C_A = 0, C_B = 0, C_C = |700 - 3475| \\ &C_D = 3 - 3, C_E = 3 - 2 \\ &C = c = 0 + 0 + 2775 * 9 + 0 + 34 * 1 = 25,009 \end{aligned} \tag{15}$$

Example 5

Referee 16 is excluded for having 0 preference for case type a, which is the case type of case8. Such assignment would violate **Hard constraint 3**. Both referees 18 and 17 have identical overall workloads and payments, however, referee 17 has a higher regional preference, which lowers his/her penalty score as follows.

$$\begin{aligned} &C_A = 240, C_B = |4240 - 4240|, C_C = |6480 - 6480| \\ &C_D = 3 - 3, C_E = 3 - 3 \\ &C = c = 240 * 16 + 0 + 0 + 0 + 0 = 3840 \end{aligned} \tag{16}$$

Example 6

Given that there are 3 cases against 4 referees, this example is more challenging in terms of average calculation of possible combination of overall pay and overall workload. The fact that none of the 4 referees violate that hard constraints makes the selection even more intriguing. Not taking into consideration *Hard constraint 1*, the optimal solution would be assigning all 3 cases to referee 3. However, this is a violation of the hard constraint as the actual workload would exceed the maximum daily working minutes of the referee. In this sense, it is not possible to assign all 3 cases to one referee. The scripted hard and soft constraints have given assign(8,19), assign(9,19) and assign(10,21) as the optimal solution. However, given the higher complexity of the manual computation of overall workload and overall payment, the optimum penalty score was not cross-validated. This applies to examples 6 - 10; therefore, the a breakdown of the optimal solution won't be included.

Example 7

Although no referee can be eliminated based on constraint violation, a few case-referee combinations are not allowed. In is particular example, case 12- referee 26 combination violates *Hard constraint 3*. Additionally, assigning referee 26 more than case violates *Hard constraint 1*.

Example 8

None of the referees or case combinations violate the hard constraints. Therefore, given that the hard and soft constraint written in ASP output the same optimal solution as expected, the optimization score was not analyzed.

Example 9

It is important to note in this example that referee 31 was not assigned a case type preference. The preference should be assigned to zero by default to correctly calculate the penalty score. This also means that referee 31 is excluded for violating *Hard constraint 3*.

Example 10

The fact that case 2 can be only assigned to referee 1 due to **Hard constraint 1**, makes the optimization process easier. However, the quality of the optimum score was not evaluated.

Conclusion

The correct optimal solutions were found for all 10 examples. However, the accuracy of the optimum score was only tested for the first 5 examples given their low level of complexity. Clashes were prominent among different problems, where making changes to find the optimal solution for one example has typically resulted in comprising the solution for another example. Weak constraints are highly dependant on hard constraints because failing to eliminate the impossible case-referee combinations will result in the wrong penalty score and consequently a non-optimal solution. In addition, it is important to eliminate case-referee combinations where either the case type preference or the case region preference of the referee is not included in the example's instance.

Opportunities for Future Work

- Since validating the optimal score was not a requirement for this project, only simple problems were manually validated. It is feasible to validate the scores for the more complex problems using a spread sheet program or a high-level programming language. It would be a value-added to the course if the CSE579 team tweaked the original ASP 2019 problem package to be tailored to ASU students as part of a peer-graded discussion to improve code quality.
- Customization of the ASP script such that it is applicable to any real-life assignment problem such as guest seating or matching a candidate to an open role. Depending on the student's interests, they can find an apt ASP application to their domain.