Insurance Referee Assignment Problem
Project Milestone 4

Rohith Danti
Arizona State University
rdanti1@asu.edu

Abstract

This report presents a refined algorithm for the Insurance Referee Assignment Problem, targeting the optimal assignment of referees to insurance cases within a single day. The algorithm balances internal efficiencies and external expense constraints, employing Answer Set Programming with Clingo to navigate complex preferences and specializations of referees. It adheres to geographical, case type, and workload capacity constraints, ensuring equitable case distribution while optimizing costs. Informed by coursework from CSE 579, this automated approach demonstrates effective management of decision-making scenarios, with potential for broader application in the insurance industry due to its demonstrated scalability and adaptability.

Problem Statement

The Insurance Referee Assignment Problem aims to efficiently allocate both in-house and external referees to assess insurance claims, considering each referee's capacity and preferences for geographic locations and incident types. The process involves assigning high-damage claims to in-house staff to handle significant risks and ensuring referees do not exceed their daily workload limits. Each claim is evaluated based on the severity of damage, influencing the compensation for external referees who are paid per case.

This complex challenge integrates decision-making and optimization to ensure that referees, qualified and conflict-free, are available to evaluate claims within specific timeframes. The assignment strategy focuses on cost efficiency, equitable workload distribution, and payment management across internal and external referees.

Specifications and Constraints

Referee Specifications: There are two types of referees: internal (salaried employees) and external (paid per case). Profiles include maximum daily workload (in minutes), preferred geographic regions, and domain preferences rated on a four-point scale.

Case Details: Cases are characterized by the claim domain, handling effort (in minutes), geographic area (postal codes), claim value (in euros), and potential compensation for external referees.

Problem Constraints: The solution must adhere to non-negotiable (hard) constraints while optimizing for cost, workload distribution, and alignment with referee preferences.

The primary objective is to develop a sophisticated algorithm that not only meets these constraints but also optimizes key performance indicators, effectively managing the allocation of referees to insurance claims.

Project background

The Insurance Referee Assignment Problem illustrates the significant benefits of integrating Knowledge Representation and Reasoning (KRR) with Answer Set Programming (ASP), utilizing Clingo as a key tool within the insurance industry. ASP's declarative programming style excels at tackling complex challenges, transforming problems into structured logic programs with clearly defined domains, constraints, and assumptions. This methodical approach consistently produces optimized outcomes, enhancing both the accuracy and operational efficiency of insurance processes.

In preparation for this project, a thorough review of academic lectures, specialized instructional videos, and extensive reference materials was conducted to build a robust understanding of ASP's core concepts, vital for crafting tailored solutions. Detailed engagement with Clingo’s documentation further refined the coding practices to align with ASP standards.

The ASP solver utilizes advanced techniques such as conflict-driven clause learning (CDCL) and backtracking to pinpoint suitable answer sets. This report examines Clingo's use of ASP, highlighting key concepts like Choice Rules, Constraints, and Optimization, among others. Clingo’s robust problem-solving capabilities are essential for tackling complex

challenges in the insurance sector. Through iterative research and application, a deep understanding of these techniques was developed, crucial for devising an optimal solution for the Insurance Referee Assignment Problem. This study underscores the effectiveness of advanced computational methods in addressing specific operational challenges within the industry.

Approach for Solving the Problem

The Referee Assignment Problem presents a multifaceted challenge that necessitates a methodical approach for resolution. In this project, Answer Set Programming (ASP) is utilized as the primary methodology for addressing the issue, with Clingo serving as the implementation tool for programming and testing the solution.

The approach to solving this problem is structured into three well-defined phases. The initial phase is dedicated to gaining a detailed understanding of the problem statement and outlining the specific solution requirements. The subsequent phase involves modeling the Referee Assignment World using Clingo, during which essential predicates are selected to accurately depict the scenario. The final phase entails coding the solution within Clingo. Each of these phases is elaborated upon in the sections that follow.

Comprehension of the Problem Statement

This phase focuses on grasping the core mapping challenge of the problem, where each case must be uniquely assigned to one referee, who may handle multiple cases. This understanding leads to the identification of essential "hard constraints" such as workload limits, regional preferences, domain expertise requirements, and claim thresholds. Additionally, "weak constraints" are identified to optimize aspects like cost efficiency, payment distribution, workload balance, and preference alignment, all varying in importance and impact on the solution's effectiveness.

Modeling the Referee Assignment World

During this phase, the scenario is accurately represented in Clingo. Predicates are categorized into three types:

Predicates Defined as Facts: These include essential data inputs such as cases and referee attributes.

Auxiliary Predicates: These offer deeper insights for understanding and solving the problem, such as payment calculations and workload assessments.

Solution Predicate (Assign): This binary predicate represents the desired outcome of the program—it defines the optimal assignments of referees to cases. The solution or output is represented using the format - *assign(case_id, referee_id)*. This indicates that the case identified by *case_id* should be assigned to the referee identified by *referee_id*.

The meticulous structuring of these predicates ensures that the entire problem is represented in a format that is not only accessible to human users but also readily interpretable by the machine. This dual focus on readability and computational feasibility is crucial for effective problem-solving within ASP environments. The careful organization and definition of these elements are foundational to constructing a solution that is both efficient and aligned with the specific nuances of the Referee Assignment Problem.

Implementation

The last phase consists of implementing the solution for the problem using Clingo. This phase is structured into three primary sections: Generate, Define, and Test. Additionally, a Display section is incorporated to limit the output strictly to the solution atoms and exclude auxiliary predicates.

Generate: This section generates potential solution candidates by enforcing a constrained choice rule that ensures each case is assigned to only one referee. The specific rule used is:

```
{assign(CID, RID): referee(RID, _, _, _, _)}=1 :- case(CID, _, _, _, _, _).
```

This rule is crucial as it establishes the solution space, strictly adhering to the one-referee-per-case mapping, and effectively setting the boundaries of the solution domain.

Define: This section establishes rules for auxiliary predicates using Clingo's computational tools like the #sum aggregate and arithmetic operations (subtraction, integer division, and absolute value). These predicates provide in-depth information crucial for refining solutions and evaluating them in the Test section. Key auxiliary predicates include:

- *totalExReferee:* Counts external referees.
- *totalReferee:* Counts all referees.
- *newWorkLoad:* Sums the efforts of newly assigned cases per referee.
- *overallWorkLoad:* Combines previous and new workloads per referee.
- *avgOverallWorkLoad:* Calculates the average total workload for referees, rounded down.
- *divergenceOverallWorkLoad:* Measures workload variation among referees.
- *newExPay:* Totals payments for newly assigned cases to external referees.
- *totalNewExPay:* Sums payments for all newly assigned cases to external referees.

- *overallExPay:* Combines previous and new payments for external referees.
- *avgOverallExPay:* Computes the average payment for external referees, truncated.
- *divergenceOverallExPay:* Assesses payment disparities among external referees.
- *caseTypeMismatchFactor:* Totals mismatch penalties for case types, based on preference differences.
- *caseRegionMismatchFactor:* Totals penalties for regional preference mismatches.

These predicates are essential for constructing a detailed profile of each assignment scenario, facilitating the optimization and equitable distribution of cases among referees.

Test: This section stringently applies hard constraints to remove unsuitable solution candidates and utilizes weak constraints to emphasize the most optimal solutions.

Hard Constraints: These constraints enforce strict requirements to ensure solution compliance with predefined operational standards. An example of such a constraint prevents any referee from exceeding their maximum workload:

```
:- assign(CID, RID), referee(RID, _, MWL, _), case(CID, _, _, _, _, _), newWorkLoad(NWL, RID), NWL > MWL.
```

This directive commands the ASP solver to exclude any assignments where a referee's new workload exceeds their allowed limit. Similar rules are applied across all essential operational parameters.

Weak Constraints: These are addressed through optimization techniques, where each constraint correlates to a cost calculated from auxiliary predicates, focusing on reducing operational inefficiencies and achieving workload balance:

1. Minimize totalNewExPay to decrease overall costs.
2. Reduce divergenceOverallExPay to ensure fair payment distribution.
3. Lower divergenceOverallWorkLoad to balance workloads equitably.
4. Decrease caseTypeMismatchFactor and caseRegionMismatchFactor to align assignments more closely with referees' preferences.

The objective function for optimization is formulated as:

```
f = (16 * totalNewExPay + 7 * divergenceOverallExPay + 9 * divergenceOverallWorkLoad + 34 * caseTypeMismatchFactor + 34 * caseRegionMismatchFactor)
```

This function incorporates various cost factors weighted according to their significance. The #minimize directive in Clingo is employed to minimize this aggregated sum, ensuring the solution optimally satisfies all set constraints.

Display: The Display section is set to only output atoms from the binary predicate assign, excluding all descriptive and auxiliary predicates to streamline result presentation.

Result And Analysis

The Clingo program was thoroughly tested on a series of sample instances with varying complexity to validate the accuracy and efficiency of the proposed solution. Each scenario's results were verified against expectations set by a domain expert, confirming that the solutions were both correct and optimally efficient.

Five distinct test scenarios were executed, named instanceOne.asp through instanceFive.asp. The program consistently achieved accurate and optimal solutions, defined as reaching the desired outcome with minimal steps. The performance outcomes for these tests are summarized in the accompanying table, illustrating the program's robust capability across different complexities.

Instance	Assignment	Optimization
instanceOne.asp	assign(4,5)	2257
instanceTwo.asp	assign(5,7)	30984
instanceThree.asp	assign(6,11)	24740
instanceFour.asp	assign(7,14)	64834
instanceFive.asp	assign(8,17)	11280

Table 1: Results for the 5 simple instances.

Furthermore, the results for each of the five simple instance files are as follows:

`clingo insurance.txt instanceOne.asp`

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo
 insurance.txt instanceOne.asp
clingo version 5.4.0
Reading from insurance.txt ...
Solving...
Answer: 1
assign(4,5)
Optimization: 2257
OPTIMUM FOUND

Models       : 1
  Optimum    : yes
Optimization : 2257
Calls        : 1
Time         : 0.016s (Solving: 0.00s 1st Model: 0.00s Unsat: 0
.00s)
CPU Time     : 0.000s
```

Figure 1: Output results of InstanceOne.asp as generated by the Clingo program.

clingo insurance.txt instanceTwo.asp

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo
 insurance.txt instanceTwo.asp
clingo version 5.4.0
Reading from insurance.txt ...
Solving...
Answer: 1
assign(5,7)
Optimization: 30984
OPTIMUM FOUND

Models       : 1
  Optimum    : yes
Optimization : 30984
Calls        : 1
Time         : 0.002s (Solving: 0.00s 1st Model: 0.00s Unsat: 0
.00s)
CPU Time     : 0.000s
```

Figure 2: Output results of InstanceTwo.asp as generated by Clingo program.

clingo insurance.txt instanceThree.asp

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo
 insurance.txt instanceThree.asp
clingo version 5.4.0
Reading from insurance.txt ...
Solving...
Answer: 1
assign(6,11)
Optimization: 24740
OPTIMUM FOUND

Models       : 1
  Optimum    : yes
Optimization : 24740
Calls        : 1
Time         : 0.007s (Solving: 0.00s 1st Model: 0.00s Unsat: 0
.00s)
CPU Time     : 0.000s
```

Figure 3: Output results of InstanceThree.asp as generated by the Clingo program.

clingo insurance.txt instanceFour.asp

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo
 insurance.txt instanceFour.asp
clingo version 5.4.0
Reading from insurance.txt ...
Solving...
Answer: 1
assign(7,14)
Optimization: 64834
OPTIMUM FOUND

Models       : 1
  Optimum    : yes
Optimization : 64834
Calls        : 1
Time         : 0.010s (Solving: 0.00s 1st Model: 0.00s Unsat: 0
.00s)
CPU Time     : 0.000s
```

Figure 4: Output results of InstanceFour.asp as generated by the Clingo program.

clingo insurance.txt instanceFive.asp

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo
 insurance.txt instanceFive.asp
clingo version 5.4.0
Reading from insurance.txt ...
Solving...
Answer: 1
assign(8,17)
Optimization: 11280
OPTIMUM FOUND

Models       : 1
  Optimum    : yes
Optimization : 11280
Calls        : 1
Time         : 0.004s (Solving: 0.00s 1st Model: 0.00s Unsat: 0
.00s)
CPU Time     : 0.000s
```

Figure 5: Output results of InstanceFive.asp as generated by the Clingo program.

Conclusion

The Referee Assignment Problem within the Vehicle Insurance sector poses a sophisticated optimization challenge aimed at allocating referees to cases, balancing workload, preferences, and geographic constraints. This research introduced a robust solution employing Answer Set Programming (ASP) and executed through Clingo, achieving optimal outcomes in diverse scenarios.

Significant contributions of this study include a detailed formulation of the problem statement specific to the vehicle insurance industry, the development of a solution approach based on ASP, and the crafting of a high-performing Clingo program. The effectiveness of this approach was thoroughly tested against the given five simple instances and the scenarios outlined in the problem description document, demonstrating its practicality and effectiveness in real-world settings.

Future Work

The While the current solution effectively addresses the Referee Assignment Problem by meeting all hard constraints, there is potential for improvement in the optimization phase. This phase involves assessing each correct solution against weak constraints to determine the most optimal one. Presently, the optimization performance is not as robust as desired, particularly in complex scenarios. Consequently, a primary avenue for future work is to enhance the efficiency of the optimization algorithm to better handle intricate real-world applications.

Additionally, further research could investigate other dimensions of the referee assignment problem that this study did not cover. For example, this research primarily concentrates on assignments within a single workday, whereas a more holistic approach would consider scheduling over extended periods. Moreover, the assumption that each case is handled by only one referee may not always be applicable. In certain situations, pairing two novice referees might yield more favorable results than assigning a single expert. Future investigations could incorporate these more nuanced aspects of the problem to forge more comprehensive and effective solutions

References

- Carmine Dodaro, Philip Gasteiger. Combining Answer Set Programming and Domain Heuristics for Solving Hard Industrial Problems.
- Marcello Balduccini. Representing Constraint Satisfaction Problems in Answer Set Programming.