

Optimization Application Demo

Resource Management Optimization

Overview

This demo considers the problem of identifying labor resources that “best” match the requirements of specific jobs.

There are two aspects that make this problem difficult:

1. How to quantify the concept of a “good” match?
2. How to explore an astronomically large solution space to identify the best assignment of resources to jobs?

The main components of the solution approach used in this demo are:

- A Machine Learning (ML) model that estimates how well a resource profile (resume) matches job requirements. Specifically, the ML model computes a matching score for all resource and job combinations that estimates the quality of the match.
- A Mixed Integer Programming (MIP) model that tackles the combinatorial complexity of the problem, producing optimal or near-optimal assignments.

Integration of ML and MIP is essential to tackling this problem. Machine Learning alone cannot address the complex combinatorial nature of the problem; MIP alone would require a human to provide a large number of matching scores, making the approach impractical.

This demo is the result of a collaboration with the Recruitology Data Science team and Tere Gonzalez - a former researcher at HP Labs and now a Senior Data Scientist at Hitachi Lab.

Introduction

Large professional services companies employ

thousands of professionals to deliver a wide variety of services. Examples of industries where professional service firms are commonly used include:

- Information technology outsourcing
- Business process outsourcing
- Accounting
- Legal
- Advertising
- Engineering
- And other specialized services

The problem of assigning appropriate people to projects is difficult by itself. Added complexity arises from the fact that professional services organizations tend to operate globally and must coordinate projects across organizational and geographic boundaries. A pipeline of projects at different stages of approval need to be staffed with a myriad of heterogeneous labor resources. With the standard work-breakdown-structure approach to project management, a project can be decomposed into sub-projects or jobs at an arbitrary level of granularity. A resource manager, or an HR specialist, must consider multiple attributes when matching project jobs and “White-Collar” personnel. The manual processes and tools that are commonly used to address such problems have many limitations, which often leads to poor demand fulfillment, low labor resource utilization, high project delivery costs, and poor customer satisfaction.

Resource matching optimization problem

For this application, consider a professional services organization that has a pipeline of project opportunities. Each project has a set of jobs that need to be filled by qualified labor resources; a resource needs to be qualified in terms of technology (Java, SQL, etc.), role

Optimization Application Demo

Resource Management Optimization

(manager, architect, tester, etc.), and domain (Finance, Marketing, etc.). Each job needs to be filled by only one qualified resource, and each qualified resource can only fill one job.

The goal is to find the “best” assignment of resources to jobs while satisfying the requirements of each job and respecting the limits on the availability of resources.

If the requirements of a job cannot be satisfied with internal resources, then new resources can be hired (for this demo, we assume that there's a limit to the number that can be hired). Some jobs may go unfilled (we will refer to these as gaps). Jobs have a priority that can be low, medium, and high. The priority of jobs determines whether we can hire new resources when those jobs cannot be filled by internal resources.

In addition, for each job, there is a minimum qualification level required to consider a resource to be qualified to perform a job (i.e., a minimum matching score).

In some cases, some resources may not be assigned (we will refer to them as idle).

In practice, this problem is typically solved either by hand or using a set of labor-intensive resource management processes and tools. A Resource Manager or an HR specialist will typically consider the requirements of a given job and try to identify a resource profile that best matches those requirements (we use ‘resource profile’ and ‘resume’ interchangeably in this demo). There are some query-based commercial tools that automate the search for someone with an appropriate resource profile, but in the end the user subjectively decides who is the best match.

There are many drawbacks of this approach:

- The manual process is time consuming
- Resources are typically assigned to jobs based on a greedy approach; an assignment is made without considering how it might affect later assignments. This often leads to poor job fulfillment and resource utilization
- Assignments are subjective and might not reflect the overall company strategy

Example of the resource matching optimization problem

To ground the ideas behind this problem, let's consider the following situation. Consider a consulting company that has three positions to fill: Tester, Java-developer, and Architect. The company has three employees available to fill the positions: Carlos, Joe, and Monika. The Resource Manager of the company estimates a matching score of resources and jobs required by a set of project opportunities. These matching scores reflect how well a person's qualifications (the resource profile in their resume) satisfy the job's requirements. Table 1 describes the matching score for each pair of resource and job, the requirements of each job, and the capacity available of each resource.

Scores	Tester	Java-Developer	Architect	Resources Capacity
Carlos	53%	27%	13%	1
Joe	80%	47%	67%	1
Monika	53%	73%	47%	1
Threshold	50%	60%	50%	
Job Requirements	1	1	1	

Table 1: Matching scores, job requirements, resource capacity and score threshold.

Optimization Application Demo

Resource Management Optimization

For example, the matching score of Monika for the Java-developer job is 73%, which means that Monika satisfies 73% of the total requirements of the Java-developer job. The score threshold for the Development job is 60%; consequently, Monika is qualified to perform this job. One Java-developer is required, and Monika has a capacity of '1' which means that she is completely available.

We might be tempted to use a simple greedy heuristic to allocate candidates to jobs. A greedy approach to tackle this problem would work as follows:

- 1) Identify the highest matching score, assign the corresponding candidate to the corresponding job, eliminate the candidate and the job from the table.
- 2) Choose the next highest score. If no more assignments are possible with scores that meet the thresholds, STOP, all jobs have been assigned to all candidates. Otherwise, go to 1.

The result of this greedy heuristic is to allocate:

- Joe to the Tester job with a matching score of 80% (> 50% job threshold)
- Monika to the Java-Developer job with a matching score of 73% (> 60% job threshold)
- Carlos to the Architect job with a matching score of 13% (< 50% job threshold, which is infeasible)

This problem is well known in linear programming and computer science literature. It is called the assignment problem. The assignment problem can be formulated as a linear programming problem and solved by the Simplex method. We should note that there's actually a more efficient algorithm, called the Hungarian method, to solve this specific problem. However, the general problem that underlies resource matching is

much more complicated than a pure assignment problem.

Typically, business conditions change frequently and an algorithm that was once efficient for solving the problem no longer works when new business conditions are considered. For the resource management problem spoken about here, the ability to hire new resources up to a certain upper limit turns out to introduce a significant complication. With this new constraint, the Hungarian method will no longer work to solve the Resource Management problem.

Hence, our demo formulates the Resource Management Optimization (RMO) problem as a Mixed Integer Programming (MIP) problem. This illustrates one of the greatest advantages of using a MIP approach to solve business problems. The new business conditions can be easily considered by adding new variables and constraints to the MIP model and then calling the Gurobi Solver to find a solution that recommends the best course of action. In contrast to a specialized assignment problem solver, a MIP approach doesn't require you to develop a new algorithm to address new business conditions.

Our RMO MIP model requires us to quantify how well a resource profile (resume) meets a job's requirements. As noted earlier, it is impractical to manually estimate these scores, so we use a Machine Learning model.

This demo makes the following assumptions:

- Each job requires only one resource
- Each resource can only fill one job
- There's no fixed (recruiting) cost associated with hiring a new resource

Optimization Application Demo

Resource Management Optimization

- The goal of the optimization model is to maximize the total matching score over all chosen assignments.
- We assume a single planning time period. More realistic models would consider multiple, dependent projects spread out over time.

Machine Learning component

We'll now take a deeper look at some of the details of the machine learning model. As previously stated, we consider three types of job attributes when evaluating a possible assignment: role, technology and domain. For each attribute, this demo only considers the following possible outcomes:

Role:	Technology:	Domain:
Developer	Web	Marketing
Data Scientist	SQL/C++	Finance
		IT/Other

The user can select a value "ANY" for each attribute to indicate that any value is acceptable for that attribute. For example, suppose you are interested in a person that knows SQL/C++, and you are indifferent to their role and domain expertise. Then, you can express this job requirement as {Role: ANY, Technology: SQL/C++, Domain: ANY}.

Our machine learning provides classifiers that are specialized for each job attribute. The results are then aggregated according to a weighting scheme into a single matching score. For a given classifier, after splitting the corpus (the set of all resumes in this case) into training (80%) and validation (20%) sets, we select a set of words that are relevant to the corresponding job attribute and characterize a resume following the

tf-idf (term frequency-inverse document frequency) approach. Next, we train the classifier and fine-tune its hyper-parameters by maximizing the performance on the validation set. We considered both naïve Bayes and random forest classifiers to see which produced better overall predictions. Note that both of these approaches perform mutually-exclusive classification. Therefore, a resume cannot be classified as Web and SQL/C++ simultaneously, even though some resources could reasonably belong to both categories.

When a new resume is to be classified, we pass it through all classifiers to compute a confidence score for each job attribute. Then, hard thresholding is performed on the scores to get a value of 1 if the score is greater than or equal to a threshold value, and 0 otherwise. Finally, a single aggregate score is computed as the weighted average of the scores over the individual attributes. For example, assume the weights of the job attributes are {Role: 45%, Technology: 35%, Domain: 20%} and let a job requirement be {Role: Developer, Technology: Web, Domain: Finance}. Furthermore, consider a resume that has been classified as {Role: Developer, Technology: Web, Domain: IT/Other}. The matching score of this resource for that job requirement is: $1*45\% + 1*35\% + 0*20\% = 80\%$.

Mixed Integer Programming (MIP) component

The main input data of the MIP model are the set of resources, the set of jobs, the maximum number of new resources that can be hired, the matching score of all resource and job combinations, and a minimum matching score for a resource to be considered for assignment for each job.

Optimization Application Demo

Resource Management Optimization

The MIP model has four types of decision variables:

- The assignment binary variables indicate when a resource has been assigned to a job.
- The hiring binary variables indicate if a new resource needs to be hired to fill the requirements of a job.
- The job gap binary variable indicates that the job could not be filled.
- The idle resource binary variable indicates that the resource has not been assigned.

The MIP model has three types of constraints:

- The demand constraints enforce that a job is either filled by an available resource, filled by hiring a new resource or not filled (thus leaving a gap).
- The resource constraints enforce the requirement that each resource is either assigned to fill a job requirement or the resource is idle.
- The hiring limit constraint ensures that the number of new resources hired is less than or equal to the hiring limit.

The objective function of the MIP model is to maximize the total matching score of the resources assigned to fill the jobs.

This simple MIP model can easily be extended in many different ways. One extension is to consider the case where the job requirement is specified in terms of FTEs (Full Time Equivalents), rather than dedicating one specific employee to each job. For example, 2.4 FTE data scientists may be required to perform a job. Also, in this extension,

we can consider that the capacity of a resource is fractional; for example, a data scientist may be available at 60% of full-time capacity. In addition, we could incorporate a fixed cost for hiring a new resource, together with a hiring budget.

Another extension is to consider resource groupings for assignments. For example, a restriction could be added that requires only resources from the same team (resource pool) to be assigned to fill the job demands of each project.

One last extension that is more realistic is to consider a multi-period case where a job's FTE requirements and a resource's fractional capacity may vary at different times in the planning horizon.

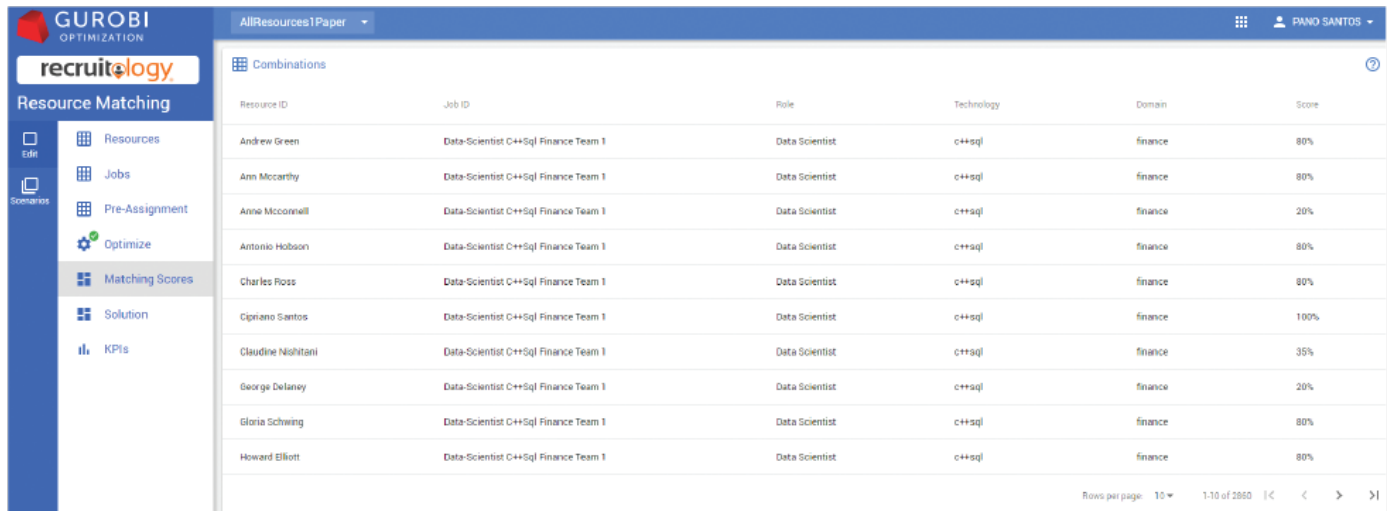
Resource Matching Optimization (RMO) Demo

We now dive into the RMO demo itself, starting with a simple scenario. This scenario considers 62 confidential resumes from the Recruitology dataset, four non-confidential resumes, and a matching number of randomly generated jobs. The weights for the job attributes are: {Role: 45%, Technology: 35%, Domain: 20%}. For this scenario, we have disabled three resources and defined one data science job as high priority with a minimum matching score requirement of 100%. Also, we allow at most one new resource to be hired.

In Figure 1, we present a screenshot of a table showing the matching scores predicted by the Machine Learning engine.

Optimization Application Demo

Resource Management Optimization



Resource ID	Job ID	Role	Technology	Domain	Score
Andrew Green	Data-Scientist C++Sql Finance Team 1	Data Scientist	c++sql	finance	80%
Ann McCarthy	Data-Scientist C++Sql Finance Team 1	Data Scientist	c++sql	finance	80%
Anne McCormell	Data-Scientist C++Sql Finance Team 1	Data Scientist	c++sql	finance	20%
Antonio Hobson	Data-Scientist C++Sql Finance Team 1	Data Scientist	c++sql	finance	80%
Charles Ross	Data-Scientist C++Sql Finance Team 1	Data Scientist	c++sql	finance	80%
Cipriano Santos	Data-Scientist C++Sql Finance Team 1	Data Scientist	c++sql	finance	100%
Claudine Nishitani	Data-Scientist C++Sql Finance Team 1	Data Scientist	c++sql	finance	35%
George Delaney	Data-Scientist C++Sql Finance Team 1	Data Scientist	c++sql	finance	20%
Gloria Schwing	Data-Scientist C++Sql Finance Team 1	Data Scientist	c++sql	finance	80%
Howard Elliott	Data-Scientist C++Sql Finance Team 1	Data Scientist	c++sql	finance	80%

Figure 1: Matching scores page.

In Figure 2, we show four tables whose content was determined by the MIP solver. The “Assignments” table displays which resources have been assigned to which jobs. In this scenario, we have 63 resources available (after disabling three), and all of them were assigned.

The “Job Gaps” table displays which jobs we could not satisfy. In this scenario, we could not fill two Data Scientist jobs. The “Job Hire” table shows jobs where we hire a new resource. In this scenario, the newly-hired resource fills a high-priority Data Scientist job, leaving gaps for three medium-priority jobs. Recall that the MIP engine will try to avoid having gaps for high priority jobs. The “Idle Workforce” table displays which resource was not assigned to any job. In this scenario, we don’t have idle resources.

In Figure 3, we display three sets of KPIs: Demand, Utilization, and Hiring.

The “Demand KPIs” encompass five metrics:

- The “Demand Fulfillment” KPI (97%) measures the percentage of the job demand that is filled with internal or hired resources.
- The “Assignment Quality” KPI (75.5%) measures the average matching score of all the assignments.
- The “Demand Fulfillment Available” KPI (95%) measures the percentage of the job demand that is filled with internal resources only.
- The “Demand Fulfillment Hiring” KPI (2%) measures the percentage of the job demand that is filled with newly-hired resources only.

Optimization Application Demo

Resource Management Optimization

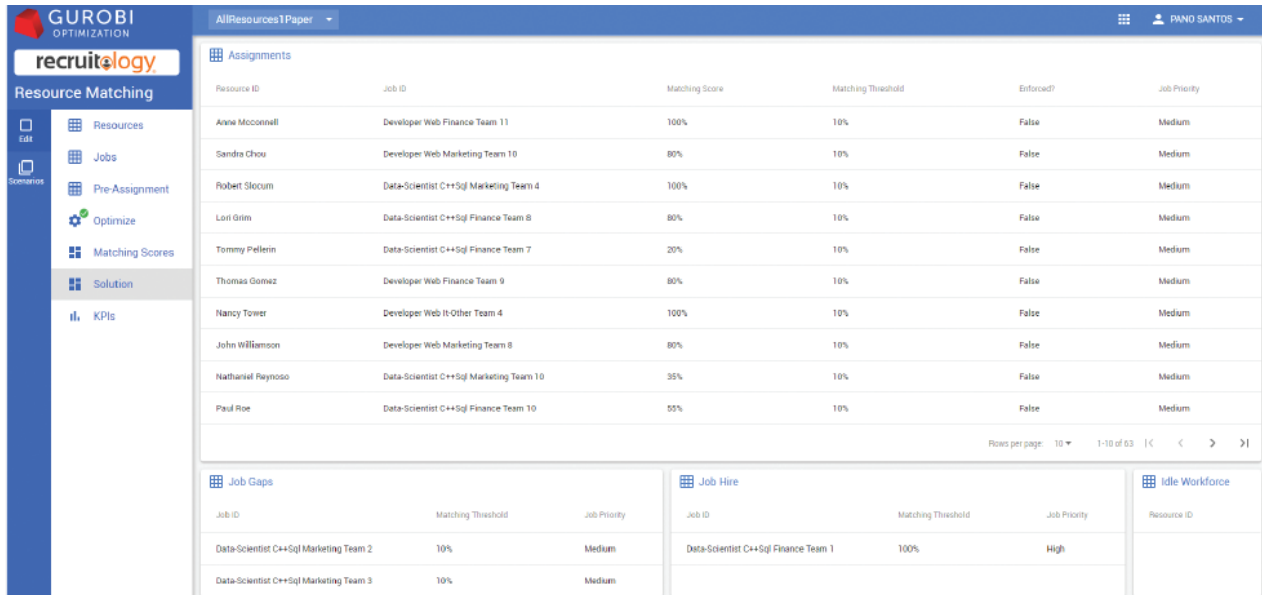


Figure 2: Solution page..

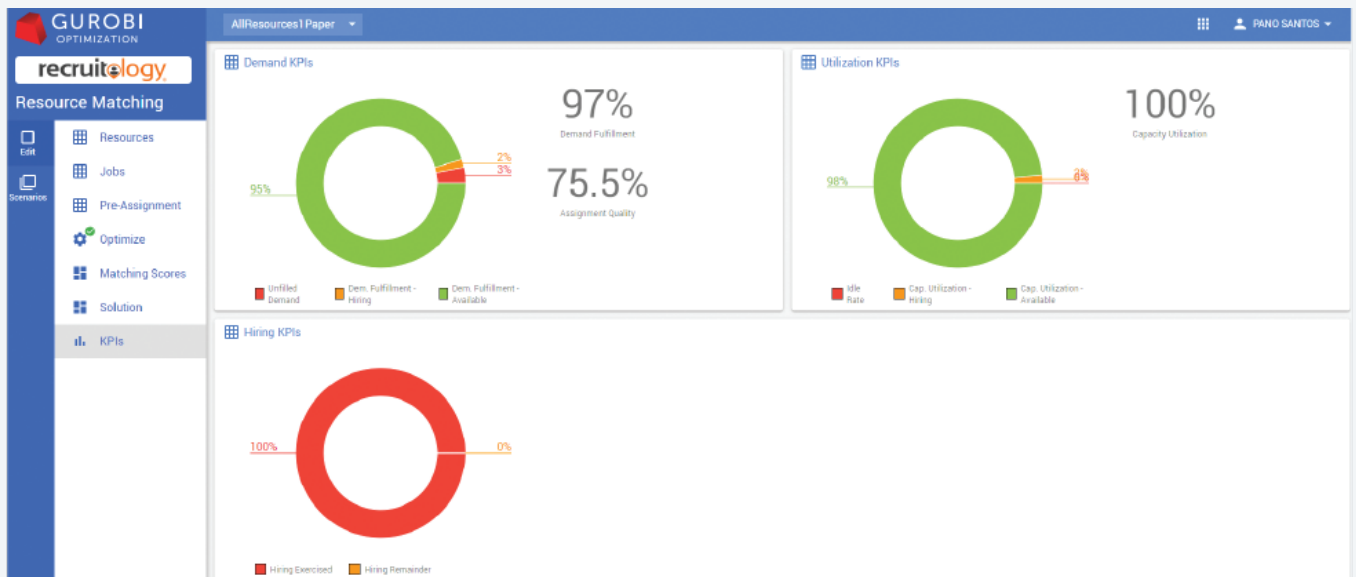


Figure 3: KPIs page

Optimization Application Demo

Resource Management Optimization

- The “Unfilled Demand” KPI (3%) measures the percentage of the job demand that could not be filled with either internal or newly-hired resources.

The “Utilization KPIs” includes four metrics:

- The “Capacity Utilization” KPI (100%) measures the percentage of available resources (internal or hired) that have been allocated to fill job demand.
- The “Capacity Utilization Available” KPI (98%) measures the percentage of the available internal resources that have been allocated to fill job demand.
- The “Capacity Utilization Hiring” KPI (2%) measures the percentage of hired resources that have been allocated to fill job demand.
- The “Idle Rate” KPI (0%) measures the percentage of the available internal resources that are idle.

The “Hiring KPIs” includes two metrics:

- The “Hiring Exercised” KPI (100%) captures newly-hired resources relative to the total number of new resources that could have been hired.
- The “Hiring Remainder” KPI (0%) measures the percentage of remaining capacity of new resources that can be hired.

Conclusion

This document discusses an optimization application demo that integrates machine learning and MIP technologies to address the fundamental problem of resource management - providing workforce resources with the right skills and capabilities, for the right job, at the right time, location, and cost. Large professional services companies employ thousands of professionals to deliver a wide variety of services, making labor the industry’s highest expense.

The problem that the RMO demo addresses is to find an assignment of resources to jobs that maximizes the total matching score of resources and jobs while satisfying the requirements of those jobs, the resource availability, and hiring constraints.

The Machine Learning component predicts the matching scores of resources and job combination while the MIP component recommends an optimal assignment of resources to jobs and an optimal hiring plan. The RMO demo computes demand fulfillment, resource utilization, and hiring KPIs that allow a Resource Manager or an HR specialist to implement the recommendations of the application or create a new scenario that may improve the KPIs.