CASE STUDY

MemComputing

Optimizing Helicopter Transportation to Oil Rigs

A major logistical problem in the Oil and Gas industry is scheduling helicopters to transport its employees to offshore drilling platforms. MemComputing delivers a solution that addresses the problem at scale and provides significant cost savings.

Introduction

The Oil and Gas industry strives to **optimize its production, reduce capital and operating costs, and maintain environmentally friendly operations**. Optimization tools are widely used across various divisions to achieve these goals, as they allow for better decision making and improved operational efficiencies. However, **there is room for significant improvement** as the scale of the mathematical equations behind many of these optimization problems are too complicated for classical computing techniques. Therefore, these companies break up the equations and rely on heuristics or simplifying algorithms, which results in an approximate or less than optimal solution. The result leads to unavoidable operational costs that can reach hundreds of millions of dollars annually per issue [1-5]. **Oil and Gas companies can positively affect the bottom line and improve their profit margins if they utilize MemComputing to address these optimization challenges.**

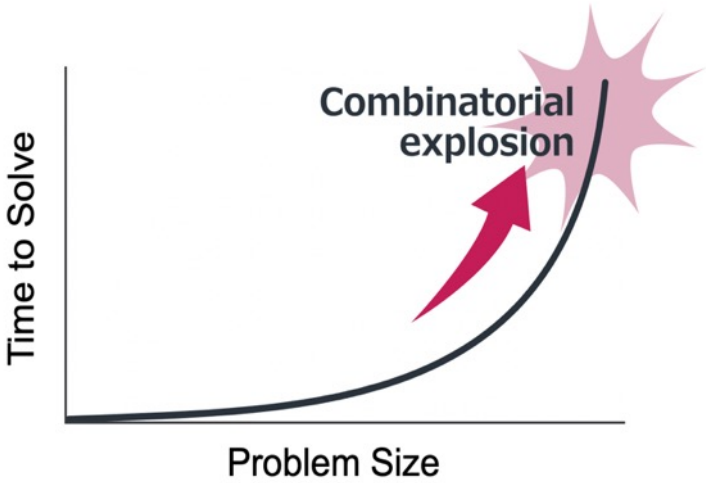
In this work, we study the logistical problem of **scheduling a fleet of helicopters for offshore deliveries to oil rigs and support vessels**. This problem is often intractable due to the number of helicopters, their different capacities, the number of offshore locations, the cargo, personnel, and time expectations of the deliveries. Here we show that **MemComputing provides a near-optimal solution in a matter of seconds** where best in class solvers produce rough approximations in hours of computation. The resulting savings in time, fuel costs, reduced maintenance, etc., can deliver **$10s to $100s of millions in savings annually** depending on the helicopter fleet's size.

The Problem

The exploration of petroleum often occurs offshore using massive oil drilling facilities, platforms, and drilling ships far from the coast. Due to the distance between onshore bases and offshore platforms, **most oil companies transport their employees and minor cargo by helicopter**. Therefore, schedules are required to accommodate varying transportation requests, destinations (shore to platform, platform to platform, platform to shore), numbers of passengers, cargo, and helicopter capacities. **The goal is to find an optimal schedule** that meets all daily transportation requests to minimize overall flight time, distance, cost, and flight time restrictions [1-5].

The problem is combinatorial, which means that as the number of passengers grow sequentially, the number of possible outcomes grows exponentially. The time to compute these problems scales exponentially for classical approaches, even when running on today's fastest supercomputers. A small version of this problem may only take minutes; however, each time the problem size doubles, the computing time can grow tenfold. It does not take long before the computing effort would require days, weeks, and even years to run [1-5].

Combinatorial Scale



The Problem Continued

For this problem, we used synthetic yet representative data associated with 24 platforms in the Gulf of Mexico with various ranges between and among the oil rigs.

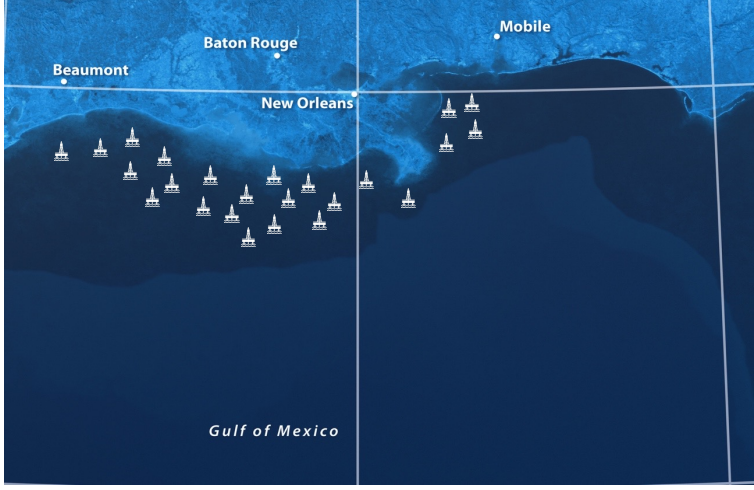


Image not to scale

There were two different types of helicopters with the following characteristics:

	Speed (knots)	Seats	Max Passenger Weight (lbs)	Max Travel Time (min)
Helicopter 1	110	6	1300	60
Helicopter 2	120	11	2200	60

We then generated random data associated with passenger schedules, starting at 40 passengers and increasing up to 300 passengers. We set unique constraints for each passenger for their weight, where they embarked, and their destination. That data looks like this:

Passenger	Embarkation	Destination	Weight (lbs)
1	Base	Rig 11	164
2	Rig 24	Base	208
3	Base	Rig 7	150
4	Rig 15	Base	243
5	Base	Rig 22	215
6	Base	Rig 15	184
7	Base	Rig 4	165
8	Rig 4	Rig 18	202
9	Rig 18	Base	175
10	Rig 18	Base	182
…	…	…	…

Traditional Methods

Many techniques have been introduced in literature and practice to address this problem. For example, Moreno et al. [1] proposed a heuristic to solve a similar situation in the Rio de Janeiro Basin. In that situation, people were manually scheduling employee travel by helicopter to the Oil Rigs (no travel between rigs). Moreno's heuristic rendered better solutions than manual schedules; the solutions averaged 15% cost and 8%-time reductions. They were then able to improve the results to this exact problem using a column generation technique [2], which breaks the problem up into multiple subproblems and solve them individually. While this improved the results over the manual solution, **the method renders an approximation and is not an optimal solution**.

Another example, Menezes et al. [3] also employed a **column generation algorithm** to create a flight planning system for Petrobras. They proposed a mathematical model to the problem based on a minimum flow network. Although they find solutions, phases are interrupted continuously, interpreted, and re-run as **the process converges very slowly and doesn't render integer solutions.**

Motta et al. [4] used a genetic algorithm to solve the Helicopter Routing Problem (HRP) to offshore platforms. The problem considered a base and five Oil Rigs. However, the author notes that **this approach was unable to generate optimal solutions.**

Rosa et al. [5] introduced **clustering search metaheuristics** to solve the Capacitated Helicopter Routing Problem (CHRP). They proposed a mathematical model for personnel transportation by helicopter to, from, and between an airport and offshore platforms. The solutions showed promising results compared to CPLEX, a leading commercial solver. However, the solutions presented did not satisfy all constraints of the problem; **therefore, the solutions were not genuinely feasible.**

We'll show that MemComputing solves the entire problem in one massively parallel compute cycle and doesn't experience exponential scaling in time. We solve this specific problem for hundreds of passengers, dozens of platforms, and tens of helicopters while satisfying all constraints. Note that the larger and more complex this problem becomes, the better MemComputing performs. As validated in our peer-reviewed paper (Traversa 2018), MemComputing demonstrates an exponential speed-up when solving hard combinatorial optimization problems.

MemComputing's Approach

MemComputing represents an entirely new way to compute combinatorial optimization problems. The underlying technology behind MemComputing is a new circuit design that produces **computational memory**, where memory performs both the task of processing and storing information [6,7]. **This new compute paradigm bypasses classical computing bottlenecks and delivers optimal solutions to currently intractable problems** [6-8]. The MemComputing circuit is so efficient that the MEMCPU™ Platform, an emulation of the circuit, is already orders of magnitude faster than today's best solutions and solves these problems in minutes. The MEMCPU Integrated Circuit, when produced, will solve these same problems in microseconds.

Unlike current methods that break the problem into smaller problems, solving these problems with our MEMCPU Platform requires that one steps back and generates a mathematical model representing the entire problem. Our MEMCPU Platform solves Integer Linear Programming (ILP) problems [9]. There are many mathematical modeling techniques, as previously discussed. However, we chose ILP because it is a standard modeling method supported by many other commercial solvers.

Developing the formulation is the hardest part of solving the problem. Since these problems are intractable, it is sometimes hard to decide where to begin. However, in general, the process is as follows:

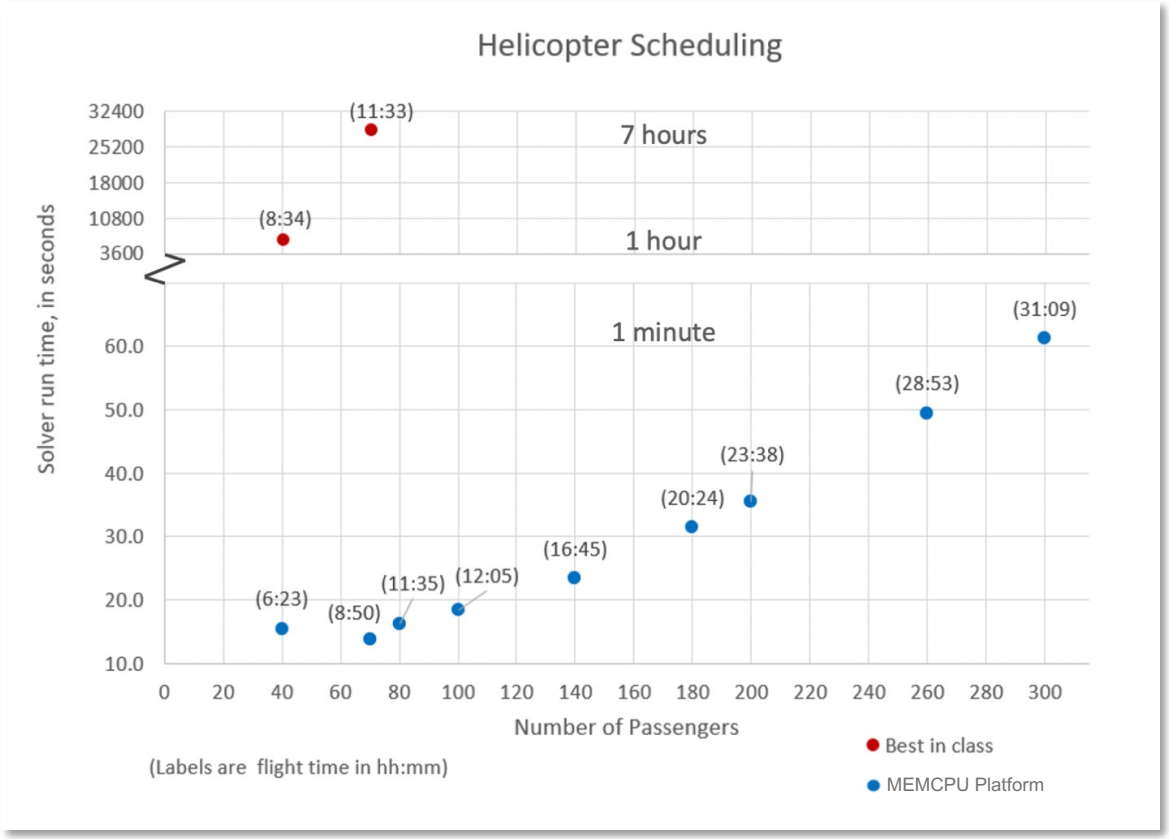
1. Identify all of the variables and related constraints.
2. Determine all objectives.
3. Develop the mathematical model in ILP
4. Test and run the problem on the MEMCPU Platform
    - 4.1 Construct the ILP model using customer data
    - 4.2 Upload the problem to the MEMCPU Platform
    - 4.3 Use the MEMCPU Platform tools to analyze the convergence and time to solution
    - 4.4 Optimize the formulation and adjust parameters as needed to improve
5. Convert the output of the MEMCPU Platform to your desired format

We offer professional services to help you with any or all of these steps if desired.
This can be done manually using the MEMCPU Platform user interface or by integrating directly using its API.

Comparisons and Results

Using Integer Linear Programming, we developed multiple problem sizes for comparison. Specifically, we generated synthetic data for problems with 40, 70, 80, 100,140, 180, 200, 260, and 300 passengers. We then ran head-to-head comparisons between the MEMCPU Platform and one of today's leading ILP solvers. The following plot demonstrates the results.

To avoid legal issues, we do not divulge the leading ILP solver that was used. However, we're happy to make the ILP problems available if you'd like to run your own comparisons.



The x-axis represents the number of passengers, and the y-axis shows the time it took to solve the problem. Next to each point in parenthesis are the total flight times that each solver found. **There are only two entries in the plot above for the leading ILP solver because it timed out after 8-hours on the 3rd run with only 70 passengers.** We did not attempt larger runs using this solver because the problem became intractable for it.

As the problem grows in size and complexity, **the MEMCPU Platform delivers near-optimal solutions and scales polynomially**. As we can see, the MEMCPU Platform solves both the small and large instances of the problem very quickly, within seconds. It finds a near-optimal solution to the largest example of the problem (300 passengers) in just 60 seconds!

Comparisons Continued

This table compares the first three runs and showing the compute time and solutions generated by the MEMCPU Platform and the Best-in-Class solver.

# of Passengers	MEMCPU Platform		Leading Solver		Improvement
	Compute Time (HH:MM:SS)	Flight Time (HH:MM:SS)	Compute Time (HH:MM:SS)	Flight Time (HH:MM:SS)	
40	00:00:15	06:22:48	01:48:53	08:33:36	25%
70	00:00:14	08:49:48	07:56:24	11:33:36	24%
80	00:00:16	11:34:58	08:00:00 (timeout)	NO RESULT	Infinite

While the compute time required is worthy of mote, the objective is to identify the most optimal schedule, defined by the combined flight times associated with all trips. Reviewing the flight times, **the MEMCPU Platform delivers significantly more efficient helicopter scheduling**. From the table, we see that the MemComputing schedule is at least 24% better when the state-of-the-art solver can find a solution. Reduced flight times mean less fuel, less wear and tear, less pilot and ground crew time, and related fees. **These cost savings add up to millions of dollars on an annual basis**. The MEMCPU Platform opens the door to improved cost savings while also enabling companies to handle larger data sets, perform scenario planning, and quickly re-schedule due to weather or mechanical issues.

Conclusion

In this case study, we considered an **intractable yet tremendously valuable optimization problem in the oil and gas industry**. The goal was to create an optimal scheduling solution for the transportation of employees and materials to, from, and between offshore platforms by helicopter. MemComputing demonstrated the ability to dramatically reduce the time to solve this problem at the scale needed in production today. Our MEMCPU Platform converged quickly to solve all sizes of the problem while also satisfying all requirements and real-world constraints. **The solutions we presented approach the optimal solution to this problem at scale and represent schedules that are ready to be deployed to the field today.**

Note that even fractional improvements to this problem can result in significant cost savings. As seen in the study, "*Optimizing Helicopter Transport of Oil Rig Crews at Petrobras*," improving the flight time & offshore landings resulted in a 14% reduction in flight cost, ultimately leading to an annual savings of more than $20 million. **We proved that MemComputing delivers improved solutions (at least 24% better) to this problem and only takes a matter of seconds (vs. hours) to compute**. This improvement would likely result in $10s of millions in annual savings. These savings go right to the bottom line.

References

[1] L. Moreno, M. P. Aragão, O. Porto, M. Reis, *Planning offshore helicopter flights on the Campos basin*, XXXVII Simpósio Brasileiro de Pesquisa Operacional (SBPO). Gramado, Brasil, 96–108 (2005)

[2] L. Moreno, M. P. Aragão, E. Uchoa, *Column generation based heuristic for a helicopter routing problem*, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, **4007**, 219–230 (2006).

[3] F. Menezes, et al. *Optimizing helicopter transport of oil rig crews at Petrobras*. Interfaces, **40**, 408–416 (2010)

[4] A. Motta, R. Vieira, J. Soletti, *Optimal Routing Offshore Helicopter Using Genetic Algorithm*. 2011 6th IEEE Joint International Information Technology and Artificial Intelligence Conference, Chongqing, China (2011)

[5] R. d. A. Rosa, A. M. Machado, G. M. Ribeiro, G. R. Mauri, *A mathematical model and a clustering search metaheuristic for planning the helicopter transportation of employees to the production platforms of oil and gas*, Computers & Industrial Engineering, **101**, 303-312 (2016)

[6] M. Di Ventra, F. L. Traversa, Perspective: Memcomputing: Leveraging memory and physics to compute efficiently, Journal of Applied Physics, **123**, 180901 (2018)

[7] A full list of paper on peer reviewed articles on memcomputing can be found at https://www.memcpu.com/publications/

[8] F. L. Traversa, P. Cicotti, F. Sheldon, M. Di Ventra, *Evidence of Exponential Speed-Up in the Solution of Hard Optimization Problems*, Complexity, **2018**, 7982851 (2018)

[9] F. L. Traversa, M. Di Ventra, *MemComputing Integer Linear Programming*, arXiv:1808.09999 (2018)