The Multi-Pump System Combinatorial Problem: A Filtering Approach Using Genetic Algorithms

Artur Tozzi de Cantuaria Gama , Samuel Kärnell , and Liselott Ericson

Fluid and Mechatronic Systems, Linköing University, Linköping, Sweden
E-mail: artur.tozzi@liu.se

Abstract

Hybrid and fully electric heavy machinery introduce new possibilities for hydraulic system design. Due to their lower energy density compared to conventional combustion engine systems, improving hydraulic efficiency is crucial. Electro-hydraulic actuators can achieve this but often require high installed power, as each actuator must be sized for maximum demand. The multi-pump system (MPS) in this paper addresses this by allowing all hydraulic machines to serve any actuator via a network of on/off valves, reducing losses and installed power. However, its multiple degrees of freedom make optimal operation non-trivial. This paper proposes a filtering strategy using a genetic algorithm to identify efficient operating points for the MPS. Although applicable for larger systems, the results here focus on an MPS with two pumps and one actuator as an example. A quasi-static system model is introduced, which the GA uses to determine steady-state control signals that minimise power consumption. The results highlight ideal operating conditions, significantly narrowing the range of viable valve combinations and pump/motor speeds. Finally, the paper discusses the limitations of the approach and its potential extension to more complex multi-pump systems for the development of dynamic control strategies.

Keywords: Multi-pump system, optimisation, genetic algorithm, hydraulic system modelling

1 Introduction

Electrification is nowadays one of the main trends in developing more efficient mobile machinery. Fully electric or hybrid vehicles aim to achieve a lower carbon footprint by removing or downsizing the conventional combustion engine, respectively. An electrified vehicle can use the conventional hydraulic system with variable displacement pumps and throttling valves to operate its functions, but these often maintain the high losses, leading to average efficiencies of around 30% [1] for the hydraulic system. Considering that battery-powered vehicles are less energy-dense, the low efficiency becomes a limitation and alternative solutions becomes more relevant.

One approach is to implement non-centralized architectures. For example, [2] compares baseline load-sensing system with displacement control, where a single prime mover drives multiple variable displacement pumps, and an alternative using electro-hydraulic actuators. A hybrid approach that combines displacement control with a smaller load sensing system is also presented in [3]. While all solutions offer higher efficiencies, they often require either increased pump capacity, installed power, or both.

An alternative solution is provided in [4] and [5], called variable-speed drive network system. This approach utilizes fixed displacement pump/motors driven by variable-speed electric machines. By eliminating valves and short-circuiting chambers or interconnecting them through the hydraulic machines, the system reduces installed motor power, but may require specific tuning for each vehicle.

Expanding on these approaches, [6] presented a new modular design, termed the multi-pump system (MPS). This architecture employs multiple fixed displacement hydraulic machines operated by variable speed electric machines, which for simplicity will often be called e-pumps in this paper, and a set of discrete valves to connected any pump/motor port to any actuator chamber. This shared configuration aims to avoid increasing the total installed power.

1.1 Combinatorial complexity in the MPS

One challenge with this system is the extremely large number of control signal combinations. A system with two main e-pumps, two actuators, and an auxiliary pump/motor would require a total of twenty valves, leading to $2^{20} = 1,048,576$ valve combinations. A relatively coarse discretization of the e-pump speed into 20 steps further increases this number to $2^{20} \times 20^2 = 419,430,400$.

A similar challenge is addressed in [7], where the authors propose a graphical approach to identify viable flow paths and systematically reduce the combinatorial complexity. Their study on two individual metering systems demonstrates that this strategy simplifies optimization while improving energy efficiency. However, it focuses on the actuator side and does not consider the role of pump/motor control in the optimization process.

A previous study by the authors of this paper conducted a backward calculation analysis to optimize the e-pumps control to minimize power consumption for the operation of one actuator [8]. The architectures considered different number of e-pumps, operating in two quadrants, for each chamber. Additionally, the paper described a sizing strategy for the hydraulic machines using scaling laws described in [9]. The system analysis showed that:

- Using three pumps for the larger chamber and two for the smaller one was the most efficient architecture, as the system can better distribute the flow to avoid low-efficiency operating points;
- Some operating regions provide opportunities to unevenly divide the flow between hydraulic machines to increase overall efficiency;
- The number of e-pumps speed combinations required large computational storage using matrices calculations, indicating that expanding the system to include more actuators and four-quadrant operation would be prohibitive.

To expand on the previous points, this paper presents a method applicable to various MPS architectures to determine the most relevant operating modes for the e-pumps and valves. Assuming four-quadrant operation, analytically determining the viable flow paths can become cumbersome. To address this, a genetic algorithm (GA) is employed to explore the design space of control inputs — namely, valve signals and pump/motor speeds — with the objective of minimizing power consumption or maximizing energy recuperation.

The approach uses a quasi-static model and does not consider a drive cycle or the transition costs between different operating points. Instead, as a result of the explorative aspect of a GA, the method provides a list of recommended operating modes for each pair of actuator force and speed. This significantly reduces the number of control options by indicating, for example, when to activate the second e-pump or when to regenerate energy. The primary aim of this paper is to demonstrate this approach for an intuitively understandable system, with the broader goal of extending it to more complex configurations.

Section 2 describes the combinatorial problem in more detail and establishes the main goals of the quasi-static model and optimisation algorithm. The modelling strategy and the GA method are detailed in sections 3 and 4, respectively. The simulation results for an MPS with two pump/motors and one actuator is presented in section 5 to show the capabilities of this approach. Section 6 then discusses how this method can be expanded to more complex systems and how the results can support the development of dynamic control strategies for the MPS.

2 System overview and problem formulation

Figure 1 shows a multi-pump system with two fixed displacement e-pumps (P_1 and P_2) powered by electric machines with variable speed drives ($VSDs$) connected to an actuator (A_1) and an auxiliary pressure source (PS_{aux}) through a set of on/off valves.

This system minimizes throttling losses by eliminating proportional valves and instead relies on flow control through variable-speed machines. While proportional valves can assist with control, their inclusion would greatly increase the problem complexity and introduce losses that the architecture aims to avoid. Furthermore, this study focuses on steady-state conditions, rendering partially open valves irrelevant for the current analysis.

The valves are named according to the components they are connecting, e.g. V_{1A1P1A} allows flow from pump port P_{1A} to actuator chamber A_{1A}. This simplified system is used as a reference for the discussions in this paper, but the method applies, in theory, to a system with any number of components.

This design allows energy regeneration and recuperation and can be expanded. Including another actuator means adding another row of valve connections to the pump/motors, adding e-pumps requires another column of valves. A constant pressure source is considered for the auxiliary line to simplify the analysis.

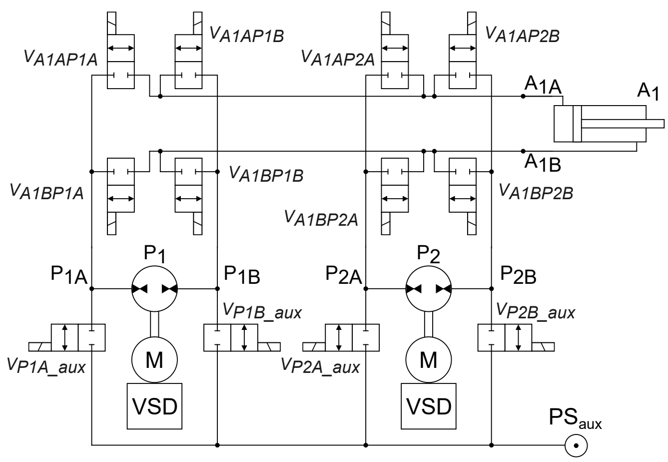

Figure 1: A diagram of a multi-pump system with one actuator controlled by two pumps with an auxiliary constant pressure source.

For the system in fig. 1, identifying valid valve combinations manually is feasible, especially because operating the hydraulic machines in four quadrants is redundant with a single actuator. Figure 2 shows three ways to achieve the same speed and force in the actuator, with each case detailed below. a) and b) show the conventional operating case with opposite pump speeds, while c) shows a regenerative case with the chambers interconnected. A similar configuration applies to motor operation.

In an operating scenario where the actuator extends against a resistive force and assuming that P_2 is off, P_1 can operate as a pump with a positive or negative speed by changing the set of open valves from V_{A1AP1A}, V_{A1BP1B}, and V_{P1B_aux} to V_{A1AP1B}, V_{A1BP1A}, and V_{P1A_aux}, respectively.

P_1 can also operate with V_{A1AP1A}, V_{A1BP1A}, and V_{P1B_aux} open, as illustrated in fig. 2 c), depending on the external force and actuator area ratio. Adding the second machine enables alternative operating modes with two pumps or one pump and a motor, for example. Since they can be controlled individually, the system can achieve the same behaviour with different combinations of angular speeds.

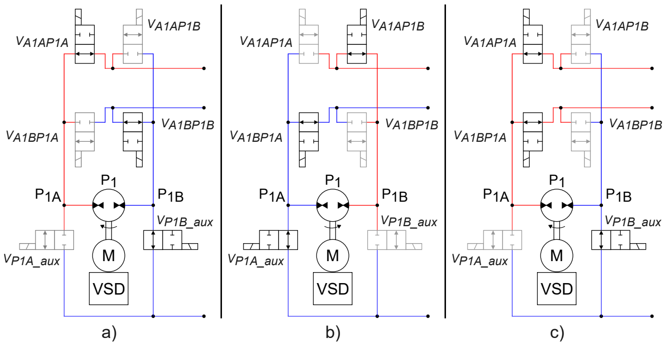

Figure 2: Operating examples for the multi-pump system in fig. 1 with a single pump. a) Pump mode with positive speed. b) Pump mode with negative speed. c) Pump mode with regeneration - chambers interconnected.

A complete system analysis would have to consider all the possible flow paths and directions to determine the optimal solution. As the number of components increases, the problem quickly becomes too complex for manual evaluation. The optimisation approach using a GA is one way to circumvent this problem for both simple and more complex systems.

2.1 Model and optimization requirements

This paper presents an approach to the aforementioned problem in two parts: a system model and an optimisation strategy to identify optimal operating conditions. The main objectives for the new model were defined as:

- Develop simplified component models that can be extended, balancing computational cost with system solvability;
- Ensure flexibility in defining the number of pumps and actuators, as the optimal architecture may vary across different applications;
- Enable both forward and backward simulation by allowing the inputs and outputs to be freely assigned, facilitating testing and results verification;
- Allow expansion of the loss models and facilitate post-processing calculation to estimate additional losses;
- Ensure smooth four-quadrant operation, eliminating the need for explicit checks on pressure difference to determine flow direction.

It was also determined that a filtering approach was needed to reduce the scope of the problem. Considering the large number of combinations, a strategy to identify the possible solutions to the system was sought after, thus this optimisation approach would need to:

- Handle a model that is non-linear and discontinuous and that can vary in size and complexity;
- Provide a reduced scope of control options to the system, thus facilitating future analysis and dynamic simulations.

The solution in this paper addresses these aspects by introducing, in sec. 3, a quasi-static model of the system based on a combination of symbolic expressions supporting multi-quadrant operation with a root-finding solver to compute the systems states. This allows the definition of different inputs and outputs for various types of simulations.

A genetic algorithm, described in sec. 4, explores the design space to find locally optimal solutions to the hydraulic machines' angular speeds and the corresponding valve signals to operate the system at steady-state conditions. The results greatly reduce the number of control options and provide insights into when the system transitions between different operating modes.

3 Quasi-static model of the hydraulic system

Based on the proposal for backward and forward simulation, the following structure is used to model the system:

- Each component may contain one or more power ports: hydraulic and mechanical. Electrical ports are omitted in this analysis;
- Ports hold the effort ($\mathcal{E}$) and flow ($\mathcal{F}$) values from one the the component's domains and share this information with other components;
- Each component includes one or more equations, including loss expressions, that define its flow and effort values;
- The communication between components is done through mechanical or hydraulic nodes, which transmit common effort values and calculate the flow balance;
- Each port is defined as a provider or receiver, thus setting the expected variable sign for the first quadrant operation. This approach handles the challenge of defining the variables signs but must be properly considered when defining the equations based on the other components.

Figure 3 illustrates a pump/motor component from fig. 1 along with its equations, connection ports, and nodes. The nodes connect to generic ports from other components, demonstrating the transmission of information. These hydraulic nodes act as abstract components that determine the flow balance between ports, mimicking the behaviour of a hydraulic volume but without accounting for dynamics.

The hydraulic flow (q) arrows show the expected positive sign for this model. The system models, for example, that the nodes receive flow from the pump/motor ports and from the ports on the bottom, and provides flows to the top ports. If the solver determines a negative value, then the flow direction is reversed. A similar consideration is performed for the valves and actuator. A more detailed explanation of the variables and equations is available in sections 3.1 and 3.4.

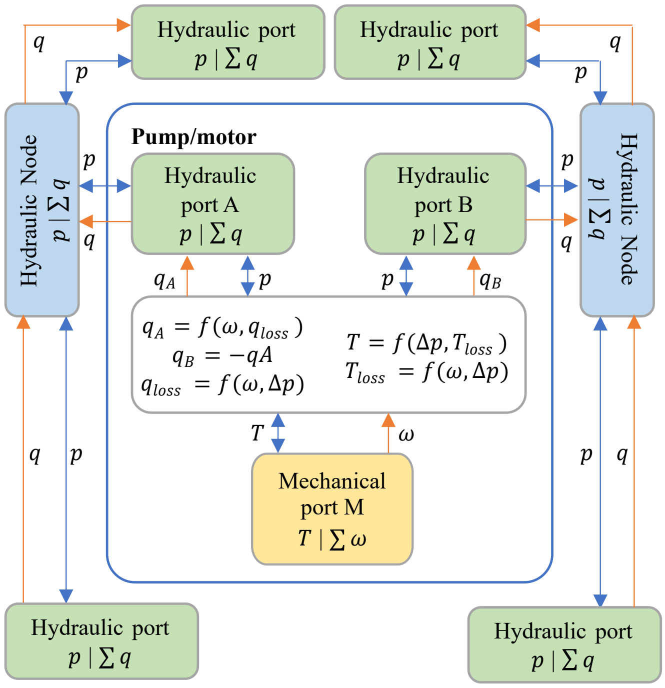

Figure 3: A diagram of a pump motor component with connection ports and nodes and the equations used to model the component.

The model assumes a positive input power at the pump/motor shaft is transmitted through the system and becomes a positive output power at the actuator rod. This means that the expected flow direction from the component ports must be taken into account in the equations, as presented in the next sections. The pressure source is an ideal component and simplifies the analysis by maintaining a constant pressure at the auxiliary line to the pump/motors.

The following subsections present the quasi-static models of pump/motors and their loss models, the linear actuator, valves, and nodes. The final subsection explains how these models are used to formulate the set of equations describing system behaviour and how variable values are provided to the solver to compute the solution.

3.1 Pump/motor model

The pump/motor model in fig. 3 considers two hydraulic ports and one mechanical port. Mechanical power going in the component, so pump operation, is assumed positive. The pump/motor is then expected to transmit positive power to the other components, so both hydraulic ports are assumed to have positive power leaving the component. The equations then explicitly define that a positive flow in one port would lead to a negative flow in the other. A summary of this logic and operating quadrants can be seen in table 1.

Table 1: Pump/motor quadrants, variable signs, and operating modes.

	T	ω	q_A	Δp	q_B	Mode
1st quad	+	+	+	+	-	Pump
2nd quad	-	+	+	-	-	Motor
3rd quad	-	-	-	-	+	Pump
4th quad	+	-	-	+	+	Motor

The equations that describe the pump/motor component are shown below, where $q_{p,A}$ is the flow at port A, D_p is the machine volumetric displacement, ω_p is the shaft angular speed, $q_{p,loss}$ is an expression for the internal leakage, T_p is the torque, Δp_p is the pressure difference between the hydraulic ports, and $T_{p,loss}$ is the torque loss.

$$q_{p,A} = D_p\omega_p - q_{p,loss}, \quad (1)$$
$$q_{p,B} = -q_{p,A}, \quad (2)$$
$$T_p = D_p\Delta p_p - T_{p,loss}. \quad (3)$$

The e-pumps' displacement vary with the number used in the system, and the sizing process follows a similar strategy to [8], with the machines sized to provide between 10% to 20% more flow than the maximum required actuator flow. The difference derives from how close the flow values are, since the new machines are sized with integer volumetric displacement. It also brings more flexibility to system operation at higher speeds.

3.1.1 4-quadrant loss models

The loss models are based on experimental data for a 32.4 cm^3/rev pump. For different machines' sizes, the losses and maximum angular speed are scaled according to the laws described in [9]. The scaling laws are shown below, where λ is the scaling factor and ω_{nom} is the nominal angular speed. The subscript ref means it is the reference pump value.

$$\lambda = \sqrt[3]{\frac{D_p}{D_{p,ref}}} \quad \Rightarrow \quad \begin{cases} \omega_{nom} = \frac{\omega_{nom,ref}}{\lambda}, \\ q_{p,loss} = \lambda^2 q_{p,loss,ref}, \\ T_{p,loss} = \lambda^3 T_{p,loss,ref}. \end{cases} \quad (4)$$

This provides loss values that cover the first quadrant of operation for the machine but to follow the model proposal, the $q_{p,loss}$ and $T_{p,loss}$ expressions should handle four quadrant operation. To cover this, the loss maps are extrapolated for the other scenarios (by inverting and mirroring the data as needed). To create the expressions, the Matlab Curve Fitting Toolbox was employed [10], generating non-linear polynomial equations to approximate the loss values as a function of the angular speed and pressure difference.

3.2 Linear actuator model

The actuator model assumes no losses and can therefore be described by the following equations, where $q_{c,A}$ and $A_{c,A}$ are the flow and area of the piston side, respectively, $q_{c,B}$ and $A_{c,B}$ are the flow and area of the rod side, v_c is the actuator speed, F_c the actuator force, $p_{c,A}$ is the pressure in the piston side and $p_{c,B}$ in the rod side:

$$q_{c,A} = A_{c,A} v_c, \tag{5}$$
$$q_{c,B} = -A_{c,B} v_c, \tag{6}$$
$$F_c = p_{c,A} A_{c,A} - p_{c,B} A_{c,B}. \tag{7}$$

3.3 Valve model

All the valves in a single architecture have the same size. They are set to have a 10 bar pressure drop at the maximum flow, determined based on the maximum actuator flow ($q_{c,max}$). This is to cover cases of energy regeneration, since all the flow of a chamber can be redirected to the other chamber of the actuator, while the pump/motor(s) compensate for the area difference.

The model assumes no leakage; therefore, the flows $q_{v,1}$ and $q_{v,2}$, representing the flows in the pressure port and the actuator port, respectively, are determined by a linear flow coefficient C_v and the pressure difference Δp_v. The variable S_v denotes the control signal, which can take binary values of 0 or 1. The pressure drop is the same in both flow directions.

A linear model simplifies the calculations without significantly affecting the results. The primary impact is a slight preference for lower flow through the valves, which may lead to solutions that favour using two pumps more frequently.

$$q_{v,1} = S_v C_v \Delta p_v, \tag{8}$$
$$q_{v,2} = -q_{v,1}, \tag{9}$$
$$C_v = \frac{q_{c,max}}{10e5}. \tag{10}$$

3.4 Node model

The node is a component responsible for combining the information from the ports of other components. In this analysis, they directly transmit the effort $\mathcal{E}$ variables (pressure, torque, force) to the connected ports.

The flow variables $\mathcal{F}$ (hydraulic flow, linear speed, angular speed) have their signs determined directly by the components, while the node performs the summation. Thus, defining $\mathcal{F}_N$ as the the total node flow and $\mathcal{F}_\#$ as a generic representation of the other flow variables, the node equation becomes

$$\mathcal{F}_N = \sum \mathcal{F}_\#. \tag{11}$$

3.5 Symbolic representation of the system

A *Python* program utilises these definitions to construct systems with varying numbers of pumps and actuators, generating the required valve components. The *SymPy* library [11] facilitates the construction of symbolic expressions based on the components equations. For the system in fig. 1, for example, the program generates 38 equations with 54 variables. These include all node equations and component loss expressions, therefore the value of 16 variables must be defined to solve the system.

Since some models are non-linear, the equations are converted into numerical expressions. The *SciPy* library [12] enables the use of various solvers to compute a numerical solution when the necessary parameters are provided.

This model works for backward and forward simulations based on which symbols are chosen as variables. For instance, with a fixed pressure for PS_{aux}, an external force value for F_C, and the signals values (S_v) for the valves, 14 variables are set. For forward simulation, one can define each pump angular speed $\omega_{p,\#}$ and solve to find the the actuator speed v_c. Alternatively, if the actuator drive cycle is given, it is possible to calculate both $\omega_{p,\#}$ by defining the sharing ratio between e-pumps. Figure 4 summarizes this process and show the integration with the genetic algorithm, detailed in section 4.

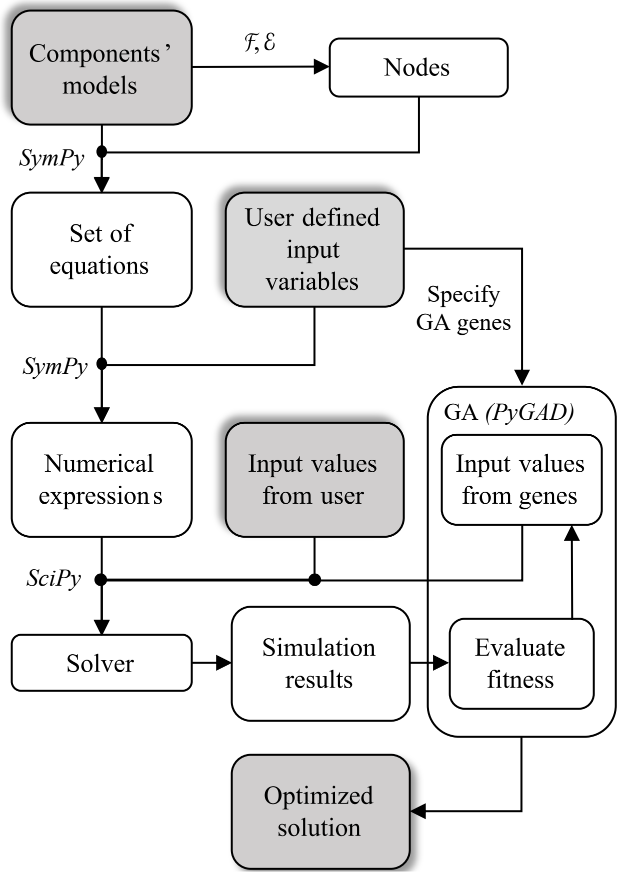

Figure 4: A diagram showing the model preparation process. Defining different input variables allow for forward and backward simulations.

4 Genetic algorithm (GA) method

As mentioned before, not all valve combinations are useful to the system, e.g. having a single valve open. In fact, many valve combinations are most likely infeasible or not useful but since they are coupled to the pump/motor speeds, it is not simple to define the available options.

Looking at the system as a control optimization problem is interesting to filter out the operating modes for the pump/motors and valves combinations. The genetic algorithm fits well into this approach because it can manage non-linear discontinuous problems and can explore a large area of the design space, thus reducing the likelihood of getting stuck in local minima.

GAs are inspired by natural evolution, where the fittest individuals are more likely to survive and pass on beneficial genes to their offspring. A chromosome is a set of genes, which, in this paper, represent the control signals to valves and pump/motors. In a simple example, two solutions (two members of the population), can provide genes to form a new chromosome to a new individual that can better fit the system's operation. To increase the variation, some genes may undergo mutation, slightly modifying the inherited information [13].

Figure 5 illustrates these steps and expands on the GA block shown in fig. 4. For each chromosome the GA creates, the genes values are sent to the solver to calculate the remaining variables. The complete results are then sent back to the GA and used to compute the fitness for each member of the population.

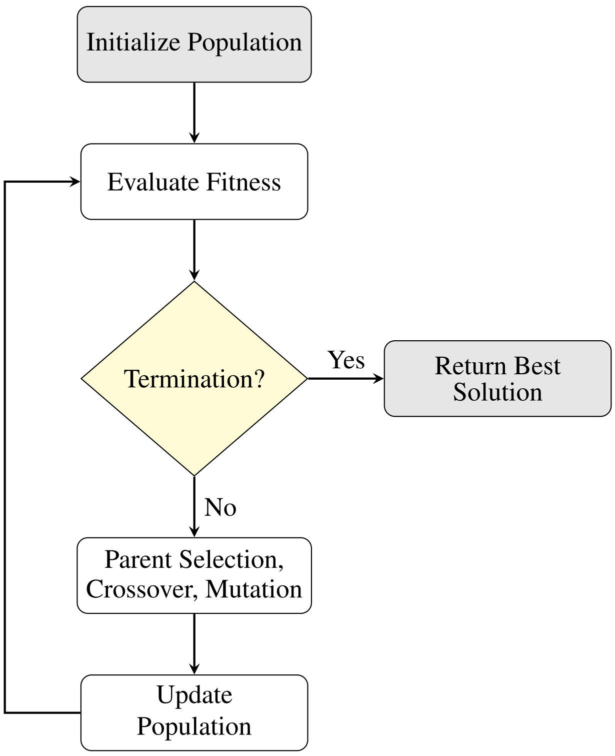

Figure 5: A simple genetic algorithm workflow

The algorithm requires a few other parameters, but the most relevant for this explanation are the population size, which determines how many individuals exist; a fitness (or cost) function that defines how good a certain individual is; and a termination criterion, which can be a fixed number of generations or another condition, such as convergence to a solution or minimal improvement over successive generations.

The reasons for choosing this method can be summarized to:

- The probabilistic approach to creating the population and offspring increases the chance of exploring distant solutions in the design space, which can converge in parallel to different local minima until the best solution is found;
- Many optimization strategies work well for continuous or discrete variables, but this system operates with binary signals for valves and continuous values for pump/motors. The GA can assign different constraints to each gene to overcome this, and the crossover step can provide large variation in values, which might lead to better exploration;

The *PyGAD* library [14] provides an easy-to-use framework for applying the algorithm in different problems. It includes standard options for gene constraints, crossover, and mutation steps, streamlining the implementation. The user is responsible for defining key parameters, such as population size, number of generations, and the fitness function.

Based on the system of equations described in the previous section, the GA can be set to create a population with any number of variables. This flexibility allows setting the valve signals and solving for the pump/motor angular speeds, or vice versa. In the more general case, both sets of values represent the genes of each chromosome. The number, types, and constraints of the genes are set from the components' parameters.

4.1 Fitness function definition

The fitness function provides a value that informs the GA on the quality of the individual. The *PyGAD* library tries to maximize the fitness value, which can combine different calculations and checks. To determine a good solution, the primary objective is to calculate the total power consumed or regenerated. The sign convention is set as: negative power represents consumption, positive power indicates regeneration.

Some constraints perform preliminary checks before calculating the fitness values. Solutions where only one valve is open can be given a value of $-inf$ to indicate infeasibility. Similar checks also exist to prevent cases where a pump/motor port is blocked if no viable path for the flow exists, if the pressure levels in the system are outside a certain range, or if a pump is mainly recirculating fluid with low Δp_p.

The total power calculation is given below, where ϕ_P is the power cost and is equal to P_{tot}, which is the total power from the pump/motors and the pressure source. n is the number of pumps, $T_{p,i}$ is the torque and $\omega_{p,i}$ is the angular speed of pump/motor i, and q_{ps} and p_{ps} are the flow rate and pressure at the pressure source, respectively. The pressure source is assumed lossless.

$$\phi_P = P_{tot} = \sum_{i=1}^{n} T_{p,i}\omega_{p,i} + q_{ps}p_{ps} \tag{12}$$

The external force applied to the system is an input that defines the pressure levels. The actuator reference speed is the target value, with the valve signals and pump/motor angular speeds as the control inputs. A cost function applies a penalty to the fitness value based on the error between the reference and actual speed, as defined below.

Here, ϕ_{v_c} represents the total penalty cost due to actuator speed errors, λ_{v_c} is a weighting factor, m is the number of actuators, and $v_{c,j,ref}$ and $v_{c,j}$ are the reference and actual speeds of actuator j, respectively.

$$\phi_{v_c} = \lambda_{v_c} \sum_{j=1}^{m} (v_{c,j,ref} - v_{c,j})^2 \tag{13}$$

The weighting factor λ_{v_c} is introduced to scale the speed penalty value to match the magnitude of the power calculation and is determined heuristically. The speed error is squared to penalize larger deviations more severely.

Other penalty functions are also implemented but are not fully described here for the sake of simplicity. They primarily perform infeasibility checks or eliminate undesirable operating cases. The final fitness value is defined as the negative sum of the preceding equations and is expressed as Φ:

$$\Phi = -(\phi_P + \phi_{v_c}) \tag{14}$$

5 Case study for a system with one actuator and two pumps

This section presents the simulation results for the system with two pumps and one actuator in fig. 1. There are relatively few solutions to the problem, so it is simple to evaluate the simulations outputs. Simulations with more pumps were also performed, while cases with multiple actuators require different visualization approaches and are not covered in this paper.

This paper uses a population size of 200 with 100 generations — values determined heuristically to balance exploration, convergence, and simulation time. The optimisation process is repeated 12 times for each actuator speed/force pair, as this repetition tends to yield more diverse solutions and reduces the likelihood of converging to poor local minima more effectively than greatly increasing the population size. Multiple solutions are also interesting when designing a control strategy as transitioning solely between the optimal operating modes determined by the GA may incur significant energy costs.

5.1 Best operating mode between the 12 iterations

Figure 6 shows which operating modes lead to the best fitness for each actuator speed and force level over the 12 iterations. "P" and "M" are pump and motor operation, respectively, and "-" means the machine is off. It is important to reinforce that this is not necessarily the global optimum, but rather the best solution found by the GA.

This graph shows the transition points between using one or more e-pumps. The machines are sized to provide, combined, more that the total flow required by the actuator, thus, in theory, a second one would only be required when the actuator speed is more than 0.125 m/s. Instead, the system starts the flow sharing earlier than that, around -0.1 and 0.1 m/s with positive external force. This indicates that the GA encounters

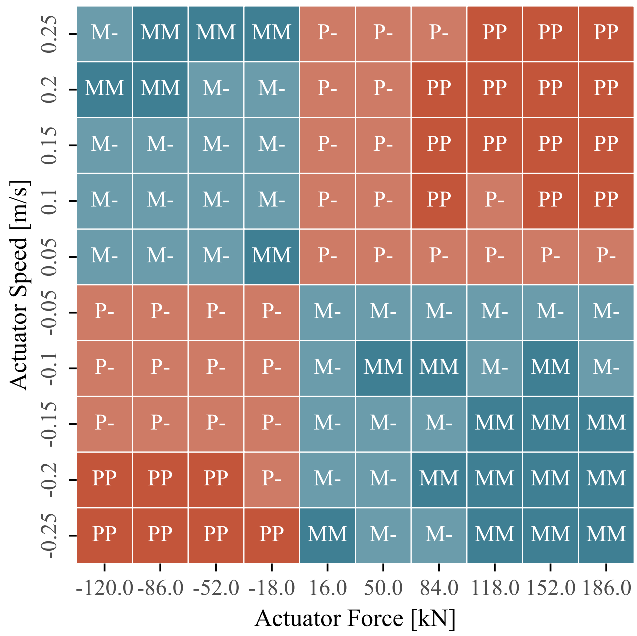

Figure 6: The best operating modes for the hydraulic machines for each actuator speed/force pair.

a region where it can minimize the system losses by using two machines at lower angular speeds.

5.2 Most frequent operating mode after 12 iterations

Defining a single operating option for the actuator's speed/force pair might not be ideal. This approach does not account for switching losses and having only one option could overly constraint the system operation. Figure 7 shows instead which operating mode most frequently happens between the 12 iterations.

The best solution available in 6 may not always be the one that most often happens. In fact, for some points the GA can determine two or more solutions that provide quite similar results just as often and with a small fitness difference, for instance around low positive forces with negative speed.

At higher speeds, both machines are typically required, as the flow demand exceeds the capacity of a single e-pump. A more interesting region is, for example, in the second quadrant, low positive speed and average negative force — table 1. From fig. 6, the best solution would be to use a single machine as a motor. However, the GA often converged to a pump-motor (PM) strategy in fig. 7. In a dynamic control context, this solution could be useful if both e-pumps were running before transitioning to this operating point, thereby avoiding the need to immediately shut down one of the machines.

5.3 Identification of regenerative cases for the best solution

Another interesting result is to analyse the pressure value in the chambers. When regeneration occurs, both actuator' chambers are connected to the same pump port, rising the pressure values. In normal operation, the low-pressure chamber

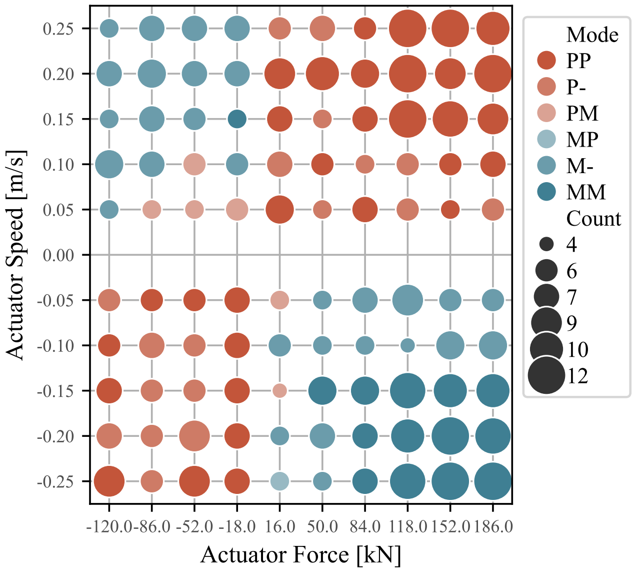

Figure 7: Most frequent operating mode for each pair speed/force.

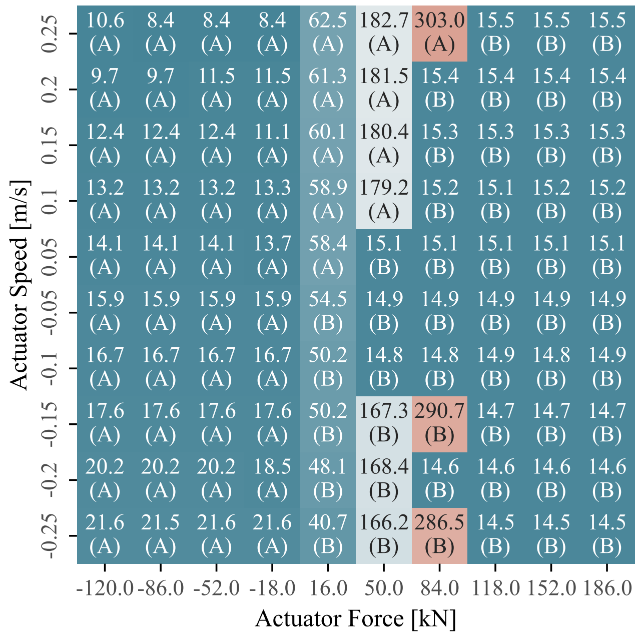

Figure 8: Low-pressure chamber values for the actuator. The higher values indicate conditions when the system is regenerating by interconnecting the chambers.

connects to PS_{aux}, thus the value varies between 10 to 20 bar, depending on flow direction and pressure drop in valves. Figure 8 shows the low pressure value for each operating point and indicates which chamber is the low-pressure side.

In the regenerative regions the average actuator pressure raises, but the e-pumps can operate at lower speeds, reducing the power cost. This seems to be the ideal case at low positive forces, and when the force increases, it looses relevance for lower speeds but stays as a reasonable option at higher absolute velocities.

The previous graphs are some examples of the information that can be extracted from the simulation results. It is also possible to identify how the flow is distributed between the e-pumps or approximate the system losses and average system efficiency at steady-state, for example.

6 Discussion and future work

Section 5 presented some results from the GA approach for a MPS with a single actuator and two pumps. The same study can be performed for a system with more e-pumps, and different constraints can be set, like limiting operation to two quadrants or focusing on a certain operating region for the actuator.

The main goal of this method was to reduce the complexity in designing a controller for a dynamic analysis. From the results, the scope of options can be greatly reduced and it provides better insights about which rules to implement. Thus, the proposal here is to use the quasi-static model to estimate the best operating modes for the system with different load conditions, and use the different options found by the GA as a guideline for the development of a dynamic control.

For example, an intuitive control strategy could start by using a single pump and adding the second machine when the required flow is too high. It is not simple to determine when to actually add this second machine, but the outcome of the GA provides a suggestion to determine the transition region between operating modes based on the actuator speed and force.

One of the challenges of using genetic algorithms arises from their probabilistic nature. Since a relatively large population is required and convergence can be slow depending on the problem, the system must be solved multiple times. A full execution of the algorithm may take days or weeks, depending on the number of components and the number of operating points set for the actuator, even with simplified models. Therefore, improvements or modifications may be necessary to apply this approach to extended architectures.

The GA could be adapted to evaluate a dynamic model and account for transition aspects during a drive cycle, but this would greatly increase the scope of the problem and constrain the results to the chosen case. Finding the ideal operating modes between two speed/force points does not guarantee optimality for the whole cycle, so the algorithm would have to account for all possible paths and compare the total energy consumption of the complete drive cycle. The probabilistic nature of the problem would most likely make it computationally too expensive.

Another common topic concerns global optimality. Dynamic programming (DP) is a well-established method often applied to analyse specific drive cycles. However, as described before, this system contains a vast number of operating points, rendering a pure DP approach computationally infeasible. The results from the GA can be used to constrain the DP algorithm by limiting the number of control options available. It is important to note, however, that this approach does not guarantee global optimality.

Nevertheless, neither strategy is feasible for implementation in an online control loop directly. The DP requires knowledge of the whole drive cycle, while the GA needs long computational time. Their main goals is to serve as a reference approach for the development and/or comparison with other control strategies which aim to mimic the results.

6.1 Future work

The results achieved here are consistent with those in [8], suggesting that the model is applicable for more complex problems. This approach is expected to be more relevant for a system with multiple actuators, as this should bring more options for system operation that are challenging to identify heuristically. Therefore, a key objective is to expand this analysis to include multiple actuators and pumps.

One constraint is that increasing the system complexity directly affects the time it takes to solve the set of equations. While memory limitation is often not a problem for GA, compared to other methods such as backward calculation or DP, computational time is a concern. Although including more actuators is feasible, adapting the algorithm to operate with parallelization could accelerate the process.

Finally, a dynamic analysis based on the results of this study can assess their practical applicability. A key question is the impact of valve switching between operating modes and the system's ability to follow the optimum operating point during different drive cycles. Additionally, GA results can support the development of real-time control strategies to support these studies.

7 Conclusions

This paper presented a filtering strategy to solve the combinatorial problem of the Multi-pump system. The large number of valves and the continuously variable speed pump/motors mean that the system can be controlled in different manners and it is not immediately clear what the viable options are. A modelling approach combined with optimisation is proposed to determine the ideal operating conditions for the system and to reduce the scope of options.

An algorithm was developed to generate symbolic quasi-static equations of the MPS architecture with varying number of pump/motors and actuators. By defining the values of certain parameters, a numerical solver is used to calculate the other variables, allowing for backward or forward simulation. A Genetic algorithm is then employed to generate possible solutions for the system and evaluate what control signals lead to minimized power consumption for different combinations of actuator speed and force.

A case study is presented for a MPS with two pumps and a single actuator as a proof of concept. The results bring insights regarding when the system should ideally transition between different operating modes for the hydraulic machines, and how to distribute the flow between them and the actuator, including regenerative and recuperative cases. This information can serve as a starting point for the development of a dynamic control to such a system.

Finally, the main interesting in the study of the MPS is to combine the operation with multiple actuators. The method proposed can be applied to extended systems, but care should be taken regarding their complexity and the time required to solve it. Nevertheless, the GA approach seems capable of handling the combinatorial problem and provides insightful information about the system behaviour that can be used for implementing dynamic control strategies.

Nomenclature

Designation	Denotation	Unit
$\mathcal{E}$	Effort variable	-
$\mathcal{F}$	Flow variable	-
q	Flow	m/^{3}/s
D_p	Volumetric displacement	m/^{3}/rad
ω	Angular speed	rad/s
T	Torque	Nm
Δp	Pressure difference	Pa
λ	Scaling factor	-
A	Area	m/^{2}
v	Linear speed	m/s^2
p	Pressure	Pa
C_v	Linear flow coefficient	m^3/s/ $\sqrt{\text{Pa}}$
S	On/off signal	-
ϕ	Cost function	-
P	Power	W
λ_{v_c}	Weight factor	-
Φ	Total fitness value	-
Subscripts		
p	Pump	
A, B, 1, 2	Port or side A, B, 1 or 2	
loss	Loss value	
nom	Nominal value	
c	Cylinder (actuator)	
v	Valve	
N	Node	
P	Power	
tot	Total	
ps	Pressure source	

Acknowledgements

This research was funded by the Strategic Vehicle Research and Innovation (FFI – Fordonsstrategisk forskning och innovation) program within the Swedish Energy Agency (Energimyndigheten) under grant number P2023-00594.

References

[1] Lauren Lynch and Bradley T. Zigler. Estimating Energy Consumption of Mobile Fluid Power in the United States. Technical report, November 2017. Report Number: NREL/TP–5400-70240, 1408087.

[2] Timir Patel, Leonardo Dos Santos Franquilino, Andrea Vacca, and Charlie Young. Comparison Study of

Fully Individualized System Architectures for Electrified Mini-Excavators: Displacement Control (DC) vs. Electro-Hydraulic Actuation (EHA). In *14th International Fluid Power Conference*, pages 22–36, Dresden, Germany, 2024. River Publishers.

[3] David Fassbender, Christine Brach, and Tatiana Minav. Energy-Efficiency Comparison of Different Implement Powertrain Concepts to Each Other and Between Different Heavy-Duty Mobile Machines. In *International Journal of Fluid Power*, pages 127–144, July 2024.

[4] Lasse Schmidt, Mikkel van Binsbergen-Galán, Reiner Knöll, Moritz Riedmann, Bruno Schneider, and Edwin Heemskerk. Energy Efficient Excavator Implement by Electro-Hydraulic/Mechanical Drive Network. *International Journal of Fluid Power*, pages 413–438, December 2024.

[5] Mikkel van Binsbergen-Galán, Barbara Videbæk, Kenneth Vorbøl Hansen, and Lasse Schmidt. Experimental Investigation of Hydraulic Power Sharing Potential in a Dual Cylinder Electro-Hydraulic Variable-Speed Drive Network. American Society of Mechanical Engineers Digital Collection, October 2024.

[6] Artur Tozzi de Cantuaria Gama, Kim Heybroek, and Liselott Ericson. A Novel Multi-pump System for Hydraulic Actuation in Electric Mobile Machinery. pages 1–7. Linköping University Electronic Press, 2023.

[7] Grégory Tardy, Eric Bideaux, Christophe Gostomski, and Armando Fonseca. Optimization Based Energy Efficient Power Transmission Design Methodology Applied to a Compact Excavator. In *GFPS PhD Symposium 2024*, Hudiksvall, Sweden, June 2024.

[8] Artur Tozzi C. Gama, Kim Heybroek, and Liselott Ericson. An Analysis of a Multi-Pump System for Actuator Operation in Electric Mobile Machinery. In *Proceedings of the ASME/BATH 2023 Symposium on Fluid Power and Motion Control*. American Society of Mechanical Engineers Digital Collection, November 2023.

[9] Damiano Padovani, Søren Ketelsen, Daniel Hagen, and Lasse Schmidt. A Self-Contained Electro-Hydraulic Cylinder with Passive Load-Holding Capability. *Energies*, 12(2):292, January 2019.

[10] The MathWorks Inc. Curve Fitting toolbox version: 24.1 (r2024a), 2024.

[11] Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. SymPy: symbolic computing in Python. *PeerJ Computer Science*, 3:e103, January 2017.

[12] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. Van Der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul Van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272, March 2020.

[13] Frederick S. Hillier and Gerald J. Lieberman. Metaheuristics. In *Introduction to operations research*, pages 635–644. McGraw-Hill, Boston, Mass., 9. ed., internat. ed edition, 2010.

[14] Ahmed Fawzy Gad. PyGAD: an intuitive genetic algorithm Python library. *Multimedia Tools and Applications*, 83(20):58029–58042, June 2024.