Proceedings of 15th International Marine Design Conference (IMDC-2024)
June 2-6, 2024
Amsterdam, the Netherlands

Operational Matrix Framework for Energy Balance Analysis for Early Stage Design of Complex Vessels

M. H. Mukti[1,*] R. J. Pawling[1], D. J. Andrews[1]

ABSTRACT

Considering vital ship systems or distributed ship service systems at the early stage of complex vessels is a challenging task. The recent UCL Network Block Approach aimed to enable ship designer to address ship systems design synthesis simultaneously as a logical network using MATLAB with a CPLEX Toolbox in MATLAB and representative piping, cabling, and trunking routings on the physical description of the ship using a proven CASD tool PARAMARINE-SURFCON. This was possible due to adopting a set of frameworks, as part of this comprehensive approach. The paper presents one of the frameworks: the Operational Matrix, to formulate distributed ship service systems network in the early stage design of complex vessels. The application of the framework could take on many forms and can be manipulated to suit a specific distributed ship service system's design. In this paper, a tutorial is given, leading from the simplest application of the Operational Matrix Framework to an example of a complex Operational Matrix application for the 3D multiplex submarine systems problem. The use of the proposed Operational Matrix Framework is shown to reveal the relationship between objective functions, constraints, bounds, and solutions of that linear programming formulation. The Operational Matrix Framework can enable the solvers (CPLEX Toolbox in MATLAB) to be very efficient in advancing early stage ship design applications. The Framework could be developed further for investigating the analysis of energy balances for new systems to achieve net zero energy demands for future naval vessels.

KEY WORDS

Distributed Ship Service Systems, Network Theory, Early Stage Ship Design.

NOMENCLATURE

Symbol	**Description**
α	Coefficient of the objective function for a cable type A
a	The set of all arcs
β	Coefficient of the objective function for a cable type B
b_i	Specific amount of commodity at a node i
$c_{i,j}$	Generic "cost" coefficient in an objective function
$\delta_{i,j}$	Binary variable of an arc i,j
e	Energy coefficient
γ_h	Power flow produced by a hub/path/intermediate node
γ_s	Power flow produced by a source node
γ_t	Power flow produced by a sink/target node
H_{sub}	Hotel load submerged for a submerged submarine
$i \in n$	A node i as a subset of a set of nodes n
$(i,j) \in a$	An arc i,j as a subset of a set of arcs a
$j \in n$	A node n as a subset of a set of nodes n
$k \in K$	An operating scenario index k as a subset of scenarios K
$\lambda_{i,j}$	Power to volume ratio of an arc i,j

[1] Marine Research Group, Department of Mechanical Engineering, University College London, London, UK; ORCID: 0000-0003-3047-1543

* Corresponding Author: hary.mukti@ucl.ac.uk

Submitted: 16 February 2024, Revised: 30 April 2024, Accepted: 1 May 2024, Published: 23 May 2024

Conference paper, DOI: https://doi.org/10.59490/imdc.2024.883 e-ISSN: 3050-486

Symbol	**Description**
$L(i,j)$	Distance between nodes i and j
$\mu_{i,j}$	Coefficient of the objective function of an arc i,j
$m \in M$	Indexed damage scenario m as a subset of scenarios M
$k \in K$	Indexed operating condition k as a subset of scenarios K
n	The set of all nodes
$OF_{i,j}$	Objective function value of an arc i,j
$P_{i,j}$	Power of an arc i,j
$u_{i,j}$	Flow capacity/variable of an arc i,j
$U_{i,j}$	Flow capacity of an arc i,j
$x_{i,j}/x_{j,i}$	Flowpath or flow variable of an arc i,j or arc j,i
$x_{i,n}$	Flow variable from a node i to node n
$x_{n,j}$	Flow variable from a node n to node j
Y_s	Power flow capacity of a source node
Y_t	Power flow capacity of a target node

1. INTRODUCTION

In the initial sizing of complex vessels, where recourse to type ship design can be overly restrictive, the estimation of the weight and space demands of vital distributed ship service systems has traditionally been poorly addressed (Andrews, 2018). The UCL Network Block Approach (Mukti, 2022) can help the concept designer to consider the impact of different configurations for the distributed ship systems in concept design with more inputs than a parametric approach but fewer assumptions than detailed systems design. This is possible because the UCL Network Block approach combines the use of an architecturally driven ship synthesis approach (Andrews, 2018), and the use of network tool called SUB/RFLOW: SUBFLOW for submarine (Mukti et al., 2021) or SURFLOW for surface ships (Mukti et al., 2024).

SUB/RFLOW was originally derived from the Architecture Flow Optimisation (AFO) (Brown, 2020) and Non-Simultaneous Multi Commodity Flow (NSMCF) (Trapp, 2015). Compared to this previous research, the SUB/RFLOW process is integrated with a 3D CASD tool (Paramarine (Qinetiq, 2019)) for designing ship systems both in terms of the network (logical representation) and the physical representation of piping, cabling, or trunking routings incorporated in the ship. The SUB/RFLOW tool uses network theory and linear programming, with nodes and arcs modelled as a set of linear programming optimisation formulations. These formulations mainly consist of constraints that define the physics of the problem and the objective function to focus the solver to maximise or minimise the solution of the problem (e.g., the multiplication between "cost" and the commodity flow) (Trapp, 2015).

The tools to solve the optimisation can use the standalone CPLEX software (IBM, 2014) or CPLEX toolbox in MATLAB (2019). The use of CPLEX toolbox in MATLAB streamlines the network analysis to be done fully in MATLAB, which allows the use of matrices for defining network properties to be read sequentially and visualised instantaneously in MATLAB. Most importantly, the use of the CPLEX toolbox in MATLAB enables the naval architect to intervene in the network formulation code for CPLEX using a MATLAB programming language (i.e., use of matrix-based computation). This, in turn, can minimise any black-box tendencies in a linear programming formulation as it reveals the interaction between the objective function, constraints, and bounds in the form of several matrices which can be themselves a single matrix, depending on the size of the network problem. Such a matrix is referred to as the Operational Matrix framework in the UCL Network Block Approach (Mukti, 2022). That framework is the focus of this paper and has been proposed to assist the designer by formulating a linear programming description, able to reflect simplified steady state temporal relationship of ship systems and the operating conditions, which has been described as the operational architecture of the ship (Brefort et al., 2018). Also, the Operational Matrix framework could be seen as a subset of the logical architecture, which interacts with a specific operating condition (e.g., loads during sprint submerged and snorting) and a discrete system's response (i.e., the simplified steady state response of specific distributed ship service systems during those particular operating conditions).

The next section of the paper provides an introduction to the proposed Operational Matrix Framework drawing on network theory. This is followed by design examples and applications of the framework to different Network Flow Optimisation setups. The Operational Matrix for the UCL Network Block Approach is then outlined. Lastly, the paper presents the advantages and limitation of the proposed framework approach and summaries what is novel in the approach through a high level comparison with previous research in the field of ship systems design.

2. THE BASIC FORM OF THE OPERATIONAL MATRIX FRAMEWORK

A network description is a set of points called nodes that are connected by lines called arcs or edges (Newman, 2010) while Network Flow Optimisation (NFO), in its basic form, is a method using linear programming that finds the lowest cost of the flow of a specific distributed commodity from a set of sources to a set of sinks (loads) through a network containing various flows (Trapp, 2015). In the case of the distributed ship service systems, the commodity can be anything that can be modelled in terms of flows, such as electrical power (energy) and data (Brown, 2020), or fluid (gas or liquid) (Trapp, 2015).

In the formulation of the linear programming of the NFO, a prescribed objective function (also called the "cost" function, which is not always the actual currency) is minimised (or maximised) by the solver, subject to constraints. The constraints define the physics of the distributed ship service systems problem in the form of equality and inequality constraints which mathematically define supply and demand limits considerations relevant to the appropriate distributed ship service systems. Thus, the units of the flow are determined and the equations describing the continuity at various supply, demand and distribution or hub nodes are derived to achieve a feasible NFO solution.

Consider a simplistic example of a NFO problem in Figure 1. This system consists of a source node, two distribution (or hub/intermediate) nodes, and a target (or sink/user) node. Described generally, an arc or edge which allows one-way flow from node i to node j is described by (i, j). Let a be the set of all arcs. Each arc (i, j) is limited to $u_{i,j}$ units of flow. There is a "cost" $c_{i,j}$ (not always currency) associated with the rate of flow in each arc of the network. Let n be the set of all nodes. Each node provides a supply or a demand of b_i units of flow. For distribution nodes, $b_i = 0$. For a supply node, $b_i > 0$ and for a demand node $b_i < 0$. The task of the solver is to find the minimum cost flow for the network such that (all) required demand is met. To do this, the solver must minimise total cost of transport over the arcs while accounting for the variation in cost for each arc (see the linear programming formulations in Table 1). To formulate as a linear programming problem, flow variables $x_{i,j}$ are used, where the annotated flow is in arc (i, j). Before there is no flow in the network, the arcs in Figure 1 are shown as thin lines. Lastly, the equation under each node represents the continuity of each node in the system as the set of constraints for the formulation.

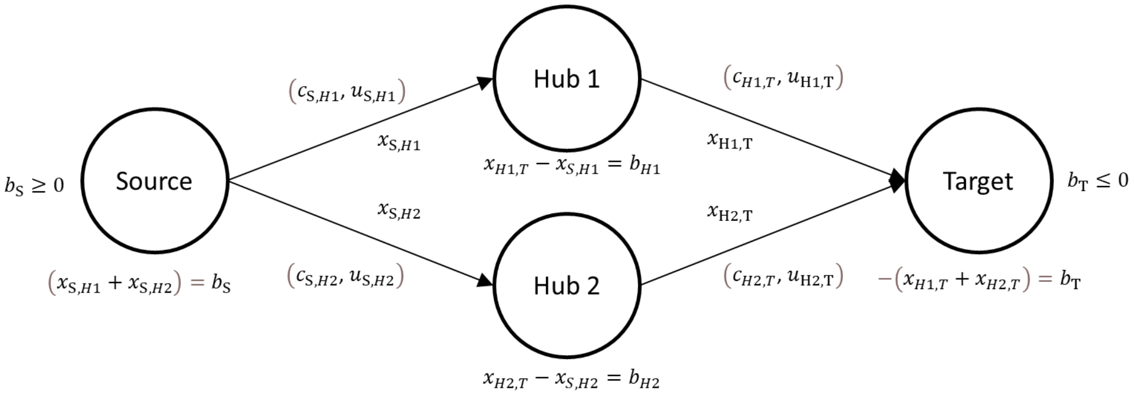


Figure 1: Simple network flow problem

Table 1: Linear programming formulations of Figure 1

Linear Programming Formulation	Mathematical Notation	Realisation
Objective Function	$min. \sum_{(i,j)\in a} c_{i,j}x_{i,j}$	$c_{S,H1}x_{S,H1} + c_{H1,T}x_{H1,T} + c_{S,H2}x_{S,H2} + c_{H2,T}x_{H2,T}$
Subject To:		
Continuity	$\sum_{(i,j)\in a} x_{i,j} - \sum_{(i,j)\in a} x_{j,i} = b_i$	$x_{S,H1} + x_{S,H2} = b_S$ $\quad x_{H1,T} - x_{S,H1} = b_{H1}$ $-x_{H1,T} - x_{H2,T} = b_T$ $\quad x_{H2,T} - x_{S,H2} = b_{H2}$
Bounds	$0 < x_{i,j} \leq u_{i,j}$	$0 < x_{S,H1} \leq u_{S,H1}$ $\quad 0 < x_{H1,T} \leq u_{H1,T}$ $0 < x_{S,H2} \leq u_{S,H2}$ $\quad 0 < x_{H2,T} \leq u_{H2,T}$
Operating Scenario	b_i	$b_S \geq 0 \quad b_{H1} = 0 \quad b_{H2} = 0 \quad b_T \leq 0$
Indices	$(i,j) \in a$	$(S, H1) \in a \; .... \; (H2, T) \in a$
	$i \in n$	$S, \ldots, T \in n$

Fundamentally, the operational matrix is used to compile all variables in the linear programming formulations into a set of rows and columns of a matrix. The matrix is called "operational" when it is specifically applied to distributed ship service systems subject to appropriate operating conditions, i.e., temporal relationships (as is demonstrated later in this paper). Thus, the operational matrix for the example in Figure 1 is shown in Table 2. The number of columns in Table 2 is given by the quantity of arcs and nodes in the network. Each row provides the "coefficients" in the linear programming formulation. In this example, it consists of the objective function or the cost function (to be minimised), the equality constraint for the continuity (i.e., all flow into and out of the node must equal the supply or demand b_i unit flow at each node), and the bounds limiting the flow to given arc capacities $u_{i,j}$ and ensuring that the flow is unidirectional. The flowpaths or flow variables $x_{i,j}$ are referred to as the set of decision variables. The solver then determines what value each of the variables in $x_{i,j}$ should take in order to minimise (or maximise, whichever is specified) the cost function. The variables in brackets for the equality constraints are not in the actual Operational Matrix. They are there to aid the ship designer's understanding of the framework.

Table 2: A simplistic example of an Operational Matrix Framework

Formulation	Arc 1	Arc 2	Arc 3	Arc 4	Source	Hub 1	Hub 2	Target
Objective function	$c_{S,H1}$	$c_{H1,T}$	$c_{S,H2}$	$c_{H2,T}$	0	0	0	0
Constraints (Equality)	$1(x_{S,H1})$	0	$1(x_{S,H2})$	0	$-1(b_S)$	0	0	0
	$-1(x_{S,H1})$	$1(x_{H1,T})$	0	0	0	$-1(b_{H1})$	0	0
	0	0	$-1(x_{S,H2})$	$1(x_{H2,T})$	0	0	$-1(b_{H2})$	0
	0	$-1(x_{H1,T})$	0	$-1(x_{H2,T})$	0	0	0	$-1(b_T)$
Lower bound	0	0	0	0	0	0	0	0
Upper bound	$u_{S,H1}$	$u_{H1,T}$	$u_{S,H2}$	$u_{H2,T}$	∞	0	0	$-\infty$

As an example, let the node and arc properties are known as summarised in Table 3 and Table 4. The network demands a 10-unit flow at the target node, so each arc is limited to 10-unit flow. For a minimisation problem, flowing through Hub 2 would be more "costly" than Hub1.

Table 3: Node properties of Figure 1

Node	Unit Flow Supply/Hub/Demand b_i	b_i value/bounds
Source	b_S	≥ 0
Hub 1	b_{H1}	0
Hub 2	b_{H2}	0
Target	b_T	-10

Table 4: Arc properties of Figure 1

Arc (i,j)	Cost $c_{i,j}$	Cost $c_{i,j}$ Value	Capacity $u_{i,j}$	Capacity $u_{i,j}$ Value
$(S, H1)$	$c_{S,H1}$	1	$u_{S,H1}$	10
$(H1, T)$	$c_{H1,T}$	1	$u_{H1,T}$	10
$(S, H2)$	$c_{S,H2}$	2	$u_{S,H2}$	10
$(H2, T)$	$c_{H2,T}$	2	$u_{H2,T}$	10

To solve the problem above, the following steps need to be undertaken:

- Generate a network matrix using an adjacency matrix or an adjacency list in MATLAB (see Table 5).
- Plug the numbers outlined in Table 3 and Table 4 into that network matrix (i.e., as network properties in MATLAB), following the format of the Operational Matrix Framework outlined in Table 2.
- The Operational Matrix can then be fed into a solver in MATLAB.
- Once the solver produces a set of flow solutions, this can be stored back in the Operational Matrix in MATLAB (see Table 6).
- Using that Operational Matrix, the system network with the network flow solution can be visualised using MATLAB.

Table 5: The adjacency matrix (left) and the adjacency list (right) of Figure 1

$$\begin{bmatrix} Node & S & H1 & H2 & T \\ S & 0 & 1 & 1 & 0 \\ H1 & 0 & 0 & 0 & 1 \\ H2 & 0 & 0 & 0 & 1 \\ T & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \begin{pmatrix} i & j \\ Source & Hub1 \\ Source & Hub2 \\ Hub1 & Target \\ Hub2 & Target \end{pmatrix}$$

The network flow solution for the problem above is shown in Figure 2.

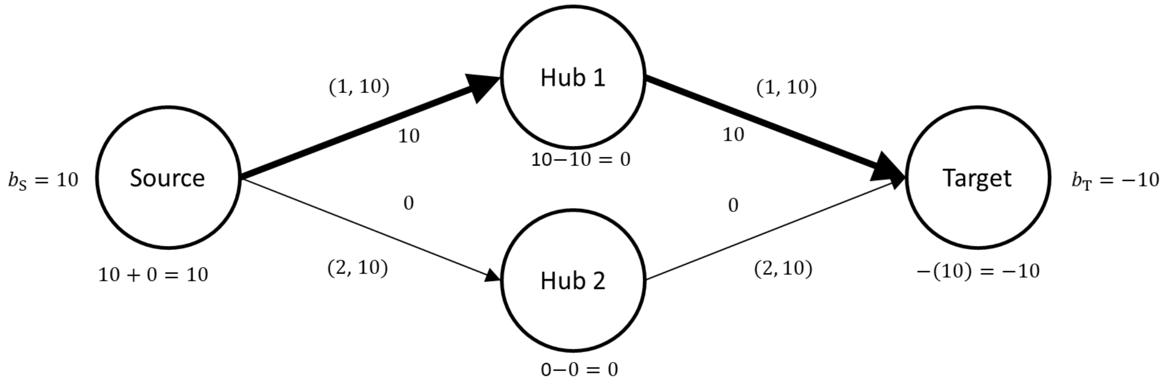


Figure 2: Simple network flow problem solution

As expected, the solver chose Hub 1 since it gives the lowest objective function value. This is 20, rather than 40, which would be the case for flowing through Hub 2. At the Source node, 10 unit flow is produced and leaving the node via arc $(S, H1)$. The continuity can be seen at the Hub 1, where 10 unit flow is entering from $(S, H1)$ and leaving the node to arc $(H1, T)$. Finally, the 10 unit flow commodity is received by the Target node from arc $(H1, T)$. The solutions satisfied the objective function (the lowest value), constraints, and the bounds. This is summarised in Table 6 using the Operational Matrix Framework (see Appendix A for the MATLAB code for this example).

Table 6: Operational Matrix Framework solution of Figure 2

Formulation	Arc 1	Arc 2	Arc 3	Arc 4	Source	Hub 1	Hub 2	Target	=
Objective function	$1(10)$	$1(10)$	$2(0)$	$2(0)$	0	0	0	0	20
Constraints (Equality)	$1(10)$	0	$1(0)$	0	$-1(10)$	0	0	0	0
	$-1(10)$	$1(10)$	0	0	0	$-1(0)$	0	0	0
	0	0	$-1(0)$	$1(0)$	0	0	$-1(0)$	0	0
	0	$-1(10)$	0	$-1(0)$	0	0	0	$-1(-10)$	0
Lower bound	0	0	0	0	0	0	0	-10	0
Upper bound	10	10	10	10	∞	0	0	-10	0

From the simple example above, the general template of the Operational Matrix Framework is given in Table 7. By using the Operational Matrix Framework, the formulation of the Network Flow Optimisation can be manipulated easily by changing the coefficients in the matrix. Understanding the use of the Operational Matrix Framework is essential before modelling, formulating, and dealing with a much larger network that could represent a large number of distributed ship service systems equipment on a vessel (Mukti, 2022). Thus, the next section provides more comprehensive examples and applications of the Operational Matrix Framework.

Table 7: The general template of the Operational Matrix framework

Formulation	Number of Arcs	Number of Nodes
Objective function	Coefficients associated with the LP formulation	
Constraints, such as equality, inequality, and bounds		

3. APPLICATIONS OF THE OPERATIONAL MATRIX FRAMEWORK

To better understand the proposed Operational Matrix Framework, this section presents three examples: the application of the Operational Matrix Framework to a simple "transportation" Non-Simultaneous Multi Commodity Flow (NSMCF) problem (Trapp, 2015); the application of the Operational Matrix Framework to a simplified Power and Propulsion Systems (PPS) submarine (SSK) problem (Mukti et al., 2021); and lastly, the application of the Operational Matrix Framework for a high-level SSK problem (Mukti, 2022).

3.1 Operational Matrix for a simple NSMCF problem

In this section, the use of an operational matrix framework for a simple NSMCF problem is presented. This is given in Figure 3 and the properties of the nodes and arcs in Table 8 and Table 9, respectively. In this example, there are two source nodes (A and E in green); two hub or intermediate nodes (C and D in black); and two user nodes (B and F in red). The top part of Figure 3 shows that the objective function result "OF" is equal to zero, as there is no flow yet in the network. Each arc has objective function "cost" $c_{i,j}$ and flow capacity $u_{i,j}$ in a form of $c_{i,j} * u_{i,j} = OF_{i,j}$. Thus, Figure 3 shows the value of $u_{i,j}$ at each arc is zero before the solver is used.

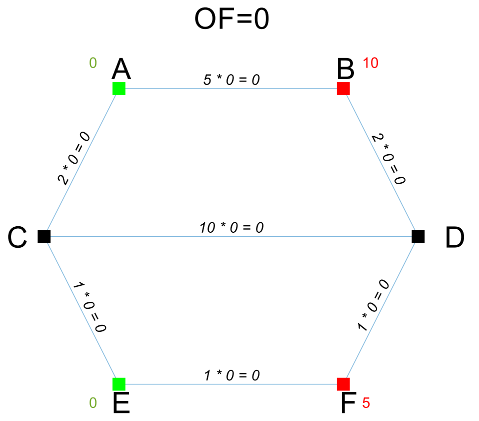


Figure 3: A simple NSMCF network problem(Trapp, 2015) coloured and revisited using the Operational Matrix framework before the NFO solver has been applied in this network

Table 8: Node properties of a simple NSCMF problem in Figure 3 so derived from Trapp (2015) node labelling, node type, and data type were added

Node n	Type	Data	Notation	b_i value/bounds
A or 1	Source	Output	b_A	≥ 0
B or 2	Target	Input	b_B	**10**
C or 3	Hub	Input	b_C	0
D or 4	Hub	Input	b_D	0
E or 5	Source	Output	b_E	≥ 0
F or 6	Target	Input	b_F	**5**

Table 9: Arc properties of the NSCMF network problem in Figure 3 connecting node *i* to node *j* outlined in Table 8 so derived from Trapp (2015)

Arc (i, j)	Cost $c_{i,j}$	Cost $c_{i,j}$ Value	Capacity $u_{i,j}$	Capacity $u_{i,j}$ Value
(A, B)	$c_{A,B}$	5	$u_{A,B}$	15
(A, C)	$c_{A,C}$	2	$u_{A,C}$	10
(B, D)	$c_{B,D}$	2	$u_{B,D}$	10
(C, D)	$c_{C,D}$	10	$u_{C,D}$	10
(C, E)	$c_{C,E}$	1	$u_{C,E}$	10
(D, F)	$c_{D,F}$	1	$u_{D,F}$	10
(E, F)	$c_{E,F}$	1	$u_{E,F}$	15

The objective function of this linear programming was to minimise the total value of the multiplication between the objective function coefficient $c_{i,j}$ and arc flow capacity $u_{i,j}$. Compared to the simple Network Flow Optimisation in Figure 1, the flowpath in this problem can be bidirectional, i.e., it can change direction (but only one direction/ non-simultaneous) in an operating condition. This means the flow variable $x_{i,j}$ can be positive or negative. Thus, inequality constraints, known as the 'capacity roll-up' (Trapp, 2015), were required to ensure the flow capacity $u_{i,j}$ is always positive as does the multiplication between the objective function coefficient $c_{i,j}$ and arc flow capacity $u_{i,j}$ regardless the sign (direction) of the flow variable $x_{i,j}$. Such a formulation is summarised in Table 10.

Table 10: The linear programming formulation and the realisation of the simple "transportation" NSMCF problem (Trapp, 2015) are presented in a table form with colour added for the application of the proposed Operational Matrix framework (see Table 11)

Linear Programming Formulation	**Mathematical Notation**	**Realisation**		
Objective Function Subject To:	$min. \sum_{(i,j)\in a} c_{i,j}u_{i,j}$	$c_{A,B}U_{A,B} + c_{A,C}U_{A,C} + c_{B,D}U_{B,D} + c_{C,D}U_{C,D} + c_{C,E}U_{C,E} + c_{D,F}U_{D,F} + c_{E,F}U_{E,F}$		
Continuity	$\sum_{(i,j)\in a} x_{i,j} - \sum_{(i,j)\in a} x_{j,i} = b_i$	$1x_{A,B} + 1x_{A,C} = 1b_A$ $-1x_{B,D} - 1x_{C,D} + 1x_{D,F} = 1b_D$	$-1x_{A,B} + 1x_{B,D} = 1b_B$ $-1x_{C,E} + 1x_{E,F} = 1b_E$	$-1x_{A,C} + 1x_{C,D} + 1x_{C,E} = 1b_C$ $-1x_{D,F} - 1x_{E,F} = 1b_F$
Capacity Rollup	$\lvert x_{i,j} \rvert \leq U_{i,j}$	$\lvert x_{A,B} \rvert \leq U_{A,B}$	$\lvert x_{A,C} \rvert \leq U_{A,C}$	$\lvert x_{B,D} \rvert \leq U_{B,D}$
		$\lvert x_{C,D} \rvert \leq U_{C,D}$	$\lvert x_{C,E} \rvert \leq U_{C,E}$	$\lvert x_{D,F} \rvert \leq U_{D,F}$
			$\lvert x_{E,F} \rvert \leq U_{E,F}$	
Bounds	$-\infty \leq x_{i,j} \leq \infty$	$-\infty \leq x_{A,B} \leq \infty$	$-\infty \leq x_{A,C} \leq \infty$	$-\infty \leq x_{B,D} \leq \infty$
		$-\infty \leq x_{C,D} \leq \infty$	$-\infty \leq x_{C,E} \leq \infty$	$-\infty \leq x_{D,F} \leq \infty$
			$-\infty \leq x_{E,F} \leq \infty$	
	$0 \leq U_{i,j} \leq u_{i,j}$	$0 \leq U_{A,B} \leq u_{A,B}$	$0 \leq U_{A,C} \leq u_{A,C}$	$0 \leq U_{B,D} \leq u_{B,D}$
		$0 \leq U_{C,D} \leq u_{C,D}$	$0 \leq U_{C,E} \leq u_{C,E}$	$0 \leq U_{D,F} \leq u_{D,F}$
			$0 \leq U_{E,F} \leq u_{E,F}$	
Operating Scenario	b_i	$b_A \geq 0$	$\boldsymbol{b_B = 10}$	$b_C = 0$
		$b_D = 0$	$b_E \geq 0$	$\boldsymbol{b_F = 5}$
Indices	$(i,j) \in a$	$(A,B) \in a .... (E,F) \in a$		
	$i \in n$	$A, \dots, F \in n$		

Using the same procedure as outlined in Section 3.1, the linear programming formulation in Table 10 is then presented as a $[23 \times 21]$ Operational Matrix and outlined in Table 11. The network solution is also included in brackets to understand the relationship between the objective function and the constraints of the linear programming formulation.

The network solution, which consists of values in brackets in the matrix, were divided into three groups based on the number of columns in Table 10. The first seven columns (black) give the flow capacity $U_{i,j}$ values, whereas the second seven columns (blue) give the flow variable $x_{i,j}$ values. The remainder values, which are in columns 15 to 20 (green or red), give the amount of supply or demand of the commodity b_i. The supply values (green) are part of the output while the demand values are part of the predefined input as shown in Table 8.

The first row of the matrix gives the objective function. The values, that are not in the bracket in the first seven columns in this row, provide coefficients $\mu_{i,j}$ for the objective function, and the remaining columns (8 to 20) were set to zero because the flow capacity $U_{i,j}$ (black) was the variable that was minimised, not the flow variable $x_{i,j}$ (blue), nor the commodity b_i (green and red).

Values at rows 2 to 7 and the first seven columns are zero because these rows were given by continuity constraints. Continuity is given by the equality constraints matrix, which consists of rows 2 to 7 and columns 8 to 21 to model six continuity constraints in Table 10. Values +1 and -1 in purple represent coefficients of continuity constraints. The realisation of 'capacity roll-up' (Trapp, 2015) that connects the flow variable $x_{i,j}$ (blue) and the flow capacity $U_{i,j}$ was applied in the inequality constraints matrix located at rows 8 to 23 and columns 1 to 14 and 21. Values -1 (dark orange) in this region indicate coefficients for the capacity roll-up (Trapp, 2015). As this formulation was applied to arcs instead of nodes, the remaining values were seen for the inequality constraints matrix, situated at rows 8 to 21 and columns 15 to 21.

Rows 22 to 23 and columns 1 to 14 show the lower bounds and the upper bounds for the flow capacity $U_{i,j}$ (black) and the flow variable $x_{i,j}$ (blue), respectively. The flow capacity $U_{i,j}$ could be used to limit the possible maximum flow capacity at each arc. However, such a formulation was not used and thus the flow capacity $U_{i,j}$ could be any positive values. Lower bounds and upper bounds that define the supply or demand amount of commodity b_i are located in the same row but in different columns, which are 15 to 20.

Arrows were added to reveal the relationship between various values and coefficients of the linear programming formulation in the matrix. Although all elements (i.e., those without bracket) in the Operational Matrix provide the input of the linear programming formulation, bounds (situated at rows 22 to 23 and columns 15 to 20) are key inputs in this formulation. Therefore, the arrows are originated from this input, which directly constrains the commodity b_i, the flow variable $x_{i,j}$, and then the flow capacity $U_{i,j}$ solutions. The flow capacity $U_{i,j}$ and the flow variable $x_{i,j}$ solutions are also constrained by bounds located at rows 22 to 23 and columns 1 to 14.

Table 11: The Operational Matrix of the basic NSCMF example, showing the simple “transportation” NSMCF example (Trapp, 2015) is presented using the proposed Operational Matrix framework with arrows showing relationships of constraints (see Table 10)

	NO	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		21
Objective Function	1	5 (0)	2 (0)	2 (10)	10 (0)	1 (0)	1 (**10**)	1 (**15**)	0	0	0	0	0	0	0	0	0	0	0	0	0		
					Arcs							Arcs						Nodes					
		$\mu_{A,B}U_{A,B}$	$\mu_{A,C}U_{A,C}$	$\mu_{B,D}U_{B,D}$	$\mu_{C,D}U_{C,D}$	$\mu_{C,E}U_{C,E}$	$\mu_{D,F}U_{D,F}$	$\mu_{E,F}U_{E,F}$	$x_{A,B}$	$x_{A,C}$	$x_{B,D}$	$x_{C,D}$	$x_{C,E}$	$x_{D,F}$	$x_{E,F}$	b_A	b_B	b_C	b_D	b_E	b_F		
					$\sum_{(i,j)\in a}\mu_{i,j}U_{i,j}$																		
												$\sum_{(i,j)\in a}x_{i,j}-\sum_{(i,j)\in a}x_{j,i}$						$-b_i$				=	0
Equality constraints matrix for continuity	2	0	0	0	0	0	0	0	1 (0)	1 (0)	0	0	0	0	0	-1 (0)	0	0	0	0	0		0
	3	0	0	0	0	0	0	0	-1 (0)	0	1 (-10)	0	0	0	0	0	-1 (-10)	0	0	0	0		0
	4	0	0	0	0	0	0	0	0	1 (0)	0	-1 (0)	-1 (0)	0	0	0	0	0	0	0	0		0
	5	0	0	0	0	0	0	0	0	0	-1 (-10)	1 (0)	0	1 (-10)	0	0	0	0	0	0	0		0
	6	0	0	0	0	0	0	0	0	0	0	0	-1 (0)	0	1 (15)	0	0	0	0	-1 (15)	0		0
	7	0	0	0	0	0	0	0	0	0	0	0	0	-1 (**-10**)	-1 (**15**)	0	0	0	0	0	-1 (-5)		0
								$-1U_{i,j}-1x_{i,j}$														≤	0
Inequality constraints matrix for bidirectionality	8	-1 (0)	0	0	0	0	0	0	-1 (0)	0	0	0	0	0	0	0	0	0	0	0	0		0
	9	0	-1 (0)	0	0	0	0	0	0	-1 (0)	0	0	0	0	0	0	0	0	0	0	0		0
	10	0	0	-1 (10)	0	0	0	0	0	0	-1 (-10)	0	0	0	0	0	0	0	0	0	0		0
	11	0	0	0	-1 (0)	0	0	0	0	0	0	-1 (0)	0	0	0	0	0	0	0	0	0		0
	12	0	0	0	0	-1 (0)	0	0	0	0	0	0	-1 (0)	0	0	0	0	0	0	0	0		0
	13	0	0	0	0	0	-1 (**10**)	0	0	0	0	0	0	-1 (**-10**)	0	0	0	0	0	0	0		0
	14	0	0	0	0	0	0	-1 (**15**)	0	0	0	0	0	0	-1 (**15**)	0	0	0	0	0	0		0
	15	-1 (0)	0	0	0	0	0	0	1 (0)	0	0	0	0	0	0	0	0	0	0	0	0		0
	16	0	-1 (0)	0	0	0	0	0	0	1 (0)	0	0	0	0	0	0	0	0	0	0	0		0
	17	0	0	-1 (10)	0	0	0	0	0	0	1 (-10)	0	0	0	0	0	0	0	0	0	0		0
	18	0	0	0	-1 (0)	0	0	0	0	0	0	1 (0)	0	0	0	0	0	0	0	0	0		0
	19	0	0	0	0	-1 (0)	0	0	0	0	0	0	1 (0)	0	0	0	0	0	0	0	0		0
	20	0	0	0	0	0	-1 (**10**)	0	0	0	0	0	0	1 (**-10**)	0	0	0	0	0	0	0		0
	21	0	0	0	0	0	0	-1 (**15**)	0	0	0	0	0	0	1 (**15**)	0	0	0	0	0	0		0
								$-1U_{i,j}+1x_{i,j}$														≤	0
Lower bounds matrix	22	0	0	0	0	0	0	0	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	0	-10	0	0	0	**-5**		
Upper bounds matrix	23	15	10	10	10	10	10	15	∞	∞	∞	∞	∞	∞	∞	∞	-10	0	0	∞	**-5**		
					$0 \le U_{i,j} \le u_{i,j}$							$-\infty \le x_{i,j} \le \infty$						b_i					

The Operational Matrix solution is visualised as a network. As shown in Figure 4, the multiplication between the "cost" coefficient $c_{i,j}$ and the flow capacity $U_{i,j}$ solution is shown as the label for each arc. Consistent with the Operational Matrix in Table 11, at the top part of Figure 4 the total objective function (OF) value is 45 (i.e., 15 from Arc (E, F) + 10 from Arc (D, F) +20 from Arc (B, D) =45).

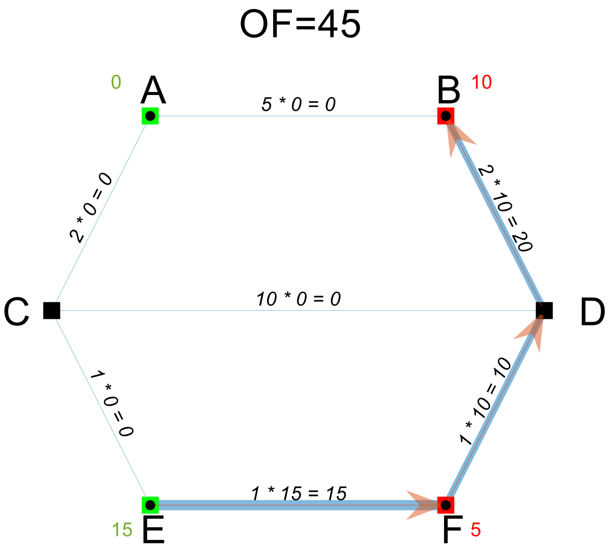


Figure 4: The network flow solution for a simple "transportation" NSMCF network problem from Trapp (2015), revisited using the proposed Operational Matrix framework

The solution shown in Figure 4 would have been different if, for example, Arc (B, D) is unavailable or damaged. This has been referred to as the minus one (M-1) survivability, which guarantees the specified demands in the network can be met with a minimum '"cost" flow although an arc is assumed to be lost (i.e., flow variable $x_{i,j} = 0$) in a given loss scenario (Trapp, 2015). Thus, if there are seven arcs, as in this example, there are seven arc loss scenarios in the linear programming formulations. Once those formulations are solved, the 'aggregate' solution, which is a term introduced by Robinson (2018), then captures maximum capacity flows in those loss scenarios i.e., an arc from the aggregate solution is sized to accommodate all possible arc flow capacities ($U_{i,j}$) in those loss scenarios. The application of such formulations in the Operational Matrix is now discussed.

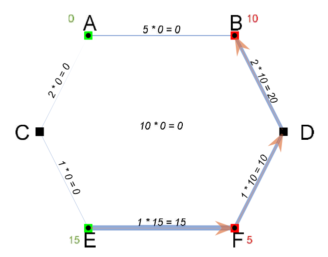
To simulate arc loss scenarios for this simple "transportation" NSMCF example, the formulation becomes multicommodity or multiflow conditions (i.e., not just one flow condition as in Figure 4). Thus, more than one set of constraints could be considered where each set of constraints represents an arc loss scenario and would be incorporated in a 'global' objective function (Trapp, 2015). This means the implementation of multiflow conditions would result in a large number of constraints in the Operational Matrix, e.g., rows 7 to 23 and columns 8 to 21 of Table 11 will expand seven fold (i.e., necessary in this example for seven arc loss scenarios). Since this expansion depends on the number of arcs a and number of nodes n in a network problem, theoretically, it can be mathematically described as a $[a^2 \times (a+n)^2]$ matrix. Hence, the scalability of the Operational Matrix for a network with (say) 100 arcs and 50 nodes would be about 10,000 rows and 22,000 columns, which would increase both the designer's workload and the solver computational resources.

Rather than expanding the Operational Matrix from that shown in Table 11, the optimisation was solved individually in each flow situation, using a loop in MATLAB. Thus, the Operational Matrix was repeated seven times (as many as the number of arcs in the network) with a flow variable $x_{i,j} = 0$ for each arc loss scenario. This can be referred to as a "single" flow formulation rather than "multi-commodity" flow formulation. The results are presented in Table 12 for each pair of networks such that the aim (left) are compared with the results of multicommodity formulation from Trapp (2015), which is given in Table 12 (right). In this comparison, some differences were found, more specifically, the flow path of scenarios (a), (c), (e), and (f), which are marked with an asterisk (*) in Table 12. These flow path discrepancies reveal that in those arc loss scenarios, the single flow formulation always gives a local minimum, i.e., the multicommodity formulation in some cases results in a higher objective function (OF) value than the single flow formulation.

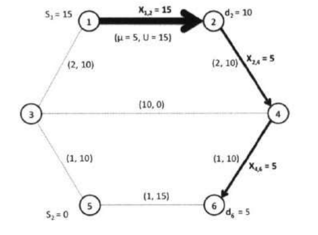
Table 12: Results comparison of a simple “transportation” NSCMF network problem between single flow formulation and multicommodity formulation, the results due to single flow formulation in the proposed Operational Matrix framework are given in the left part of each cell, while the multicommodity solution from Trapp (2015) is presented in the right part of each cell with the OF value added by the candidate at the top part of each cell (nodes A, B, C, D, E, F are equivalent to nodes 1, 2, 3, 4, 5, 6)

(a) No loss*

OF=45



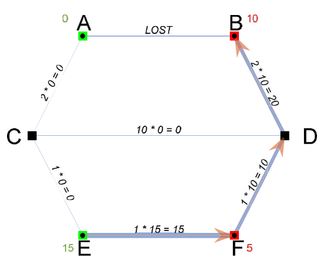
OF=90



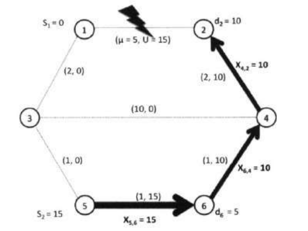
(Trapp, 2015)

(b) Loss of arc A,B

OF=45



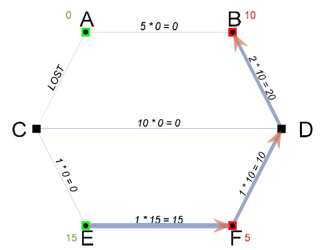
OF=45



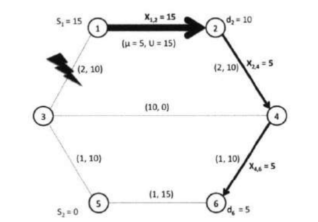
(Trapp, 2015)

(c) Loss of arc A,C*

OF=45



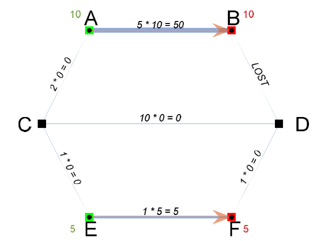
OF=90



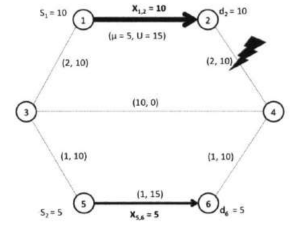
(Trapp, 2015)

(d) Loss of arc B,D

OF=55



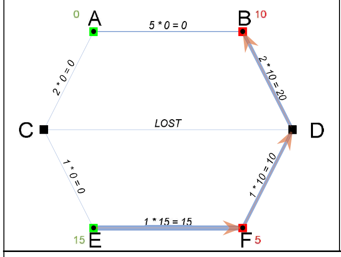
OF=55



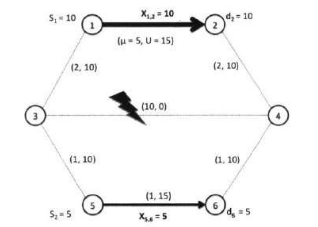
(Trapp, 2015)

(e) Loss of arc C,D*

OF=45



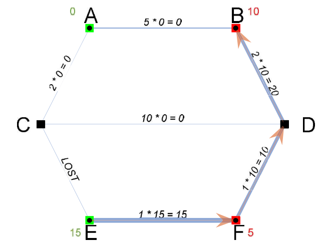
OF=55



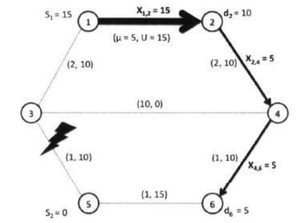
(Trapp, 2015)

(f) Loss of arc C,E*

OF=45



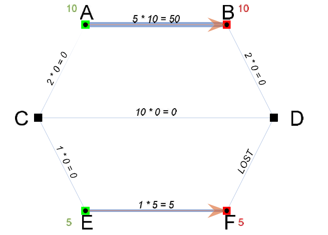
OF=90



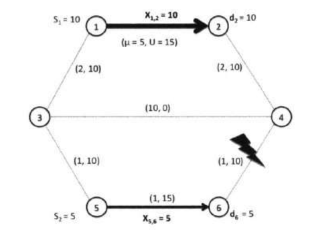
(Trapp, 2015)

(g) Loss of arc D,F

OF=55



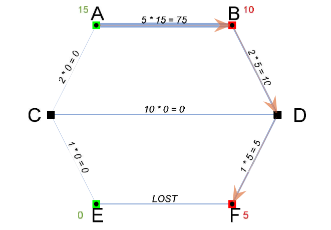
OF=55



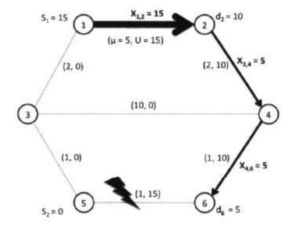
(Trapp, 2015)

(h) Loss of arc E,F

OF=90



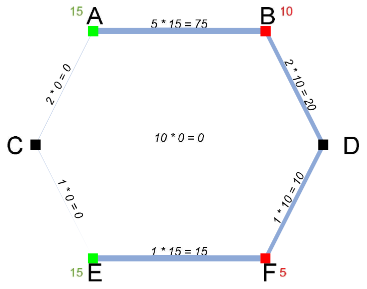
OF=90



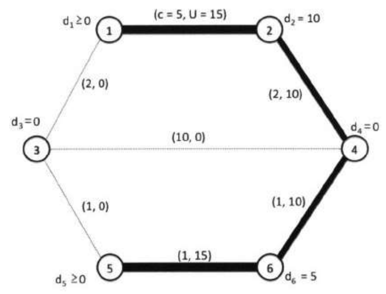
(Trapp, 2015)

Aggregate solution

OF=120



OF=120



(Trapp, 2015)

Despite the difference in terms of the local minima, the single flow formulation gives the same aggregate result as the multicommodity formulation (see the aggregate solution at the bottom part of Table 12). This confirms that the same aggregate solution in this specific NSMCF example can be obtained more efficiently with fewer constraints without the need to include all arc loss scenarios in the global objective function. Therefore, this example suggests that by using the proposed Operational Matrix Framework, the input required for the NFO could potentially be easily manipulated and reduced. This would be more efficient for quick distributed ship service systems sizing focused investigations and thus more appropriate for early-stage ship design applications.

The next section provides the application of the Operational Matrix Framework to simplified power and propulsion systems in a diesel-powered submarine.

3.2 The Operational Matrix applied to simplified submarine power and propulsion systems

This section describes the Operational Matrix Framework used to solve a simplified power and propulsion systems (PPS) SUBFLOW problem outlined in (Mukti et al., 2021). In the SUBFLOW formulation, there are only two broad types of nodes: terminal nodes and hub nodes (Mukti, 2022). Terminal nodes were used to model sources or sinks at the extremities of the flow. The extremities in the network were identified by the number of in-degree and out-degree flows. If a terminal node has only one or multiple out-degree flows (diverging), that node was taken to be a source. Conversely, if the flow(s) were converging and there were no out-degree flow(s), that node would have been considered as a sink/ target. Figure 5 shows the PPS configuration of a diesel-powered submarine (SSK) study, which was taken from a 3D Paramarine-SURFCON synthesis process (Mukti et al., 2021).

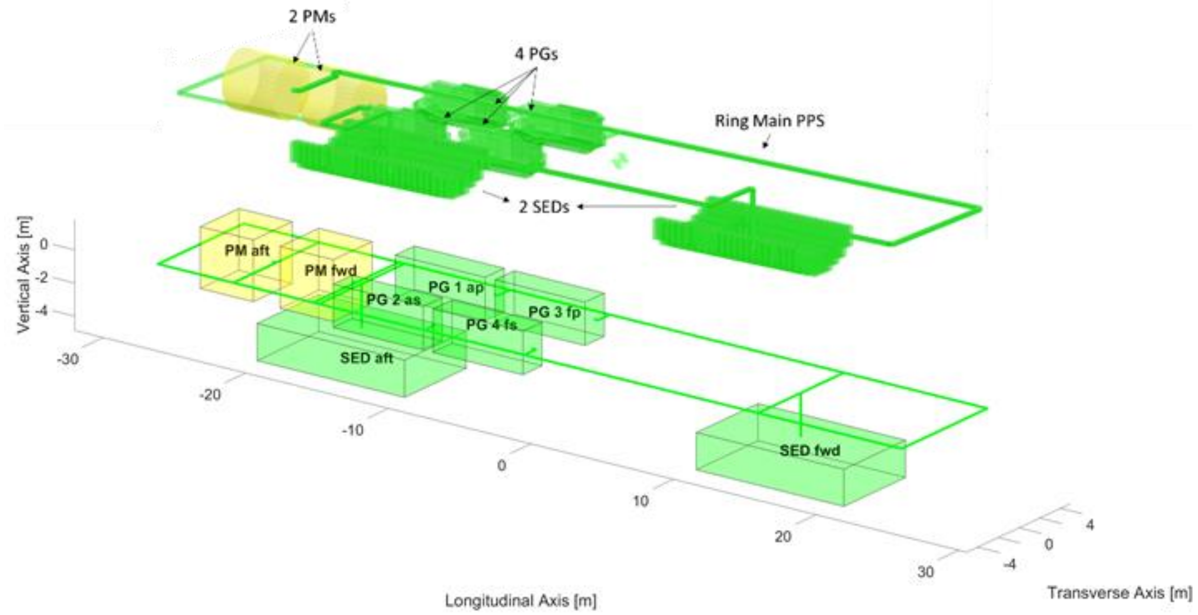


Figure 5: A simplified PPS architecture displayed in Paramarine-SURFCON (top) translated into MATLAB model for the SUBFLOW analysis (bottom) on the SSK Case Study

The simplified PPS 3D model above was then taken as a basis for the logical network as shown in Figure 6 with the network properties in Table 13. Figure 6 shows the PPS ring-main configuration network consists of 32 nodes and 36 arcs. There are two Propulsion Motors (PMs) as target (user) nodes, four Power Generations (PGs) as source nodes, and two electrical Stored Energy Devices (SEDs). The SED nodes can be the demand nodes during a snorting operating condition and can be supply nodes during a submerged operating condition. The rest of the nodes in the PPS network are hub or junction nodes. The nodes properties in Table 13 had to be defined from design requirements, i.e., the demanded power was based on the baseline SSK design (Mukti et al., 2021).

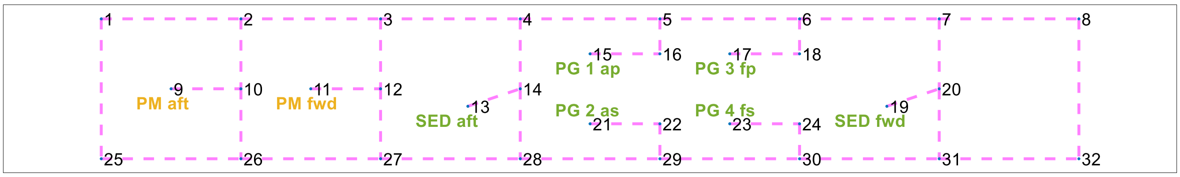


Figure 6: Nodes labelling to the PPS architecturen MATLAB (not to scale)

Table 13: The summary of the power commodity in snorting and transit

System Component	**Supply Y_s** (kW)	**Demand Y_t** (kW)	**Node ID** (Figure **6)**
PM aft	-	346	9
PM fwd	-		11
PG 1 ap	1600 (max)	-	15
PG 2 as	1600 (max)	-	21
PG 3 fp	1600 (max)	-	17
PG 4 fs	1600 (max)	-	23
SED aft	-	2930	13
SED fwd	-	2930	19

As an early development of SUBFLOW formulation as outlined in Table 14, this PPS example was kept simple (i.e., it does not represent myriad components in the actual submarine PPS). The equations are now briefly described in turn, the reasonings for the formulation are addressed in detail in (Mukti et al., 2021).

Table 14: Linear programming formulation and realisation of a simplified PPS SUBFLOW problem

Linear Programming Formulation	**Mathematical Notation**	**Realisation**
Objective Function:	$\sum_{(i,j)\in a} \left(\alpha\ \delta_{i,j} + \beta\ \delta_{i,j} + \lambda_{i,j}\ P_{i,j}\right)$	$\left(\alpha\ \delta_{1,2} + \beta\ \delta_{1,2} + \lambda_{1,2}\ P_{1,2}\right) + \cdots$ $\left(\alpha\ \delta_{31,32} + \beta\ \delta_{31,32} + \lambda_{31,32}\ P_{31,32}\right)$
Subject To		
Continuity	$\sum_{(i,j)\in a}^{k} x_{i,j}^{k} - \sum_{(i,j)\in a}^{k} x_{j.i}^{k} = \gamma_i^k$	$x_{1,2}^k - x_{25,1}^k = \gamma_1^k$ ... $x_{32,8}^k - x_{31,32}^k = \gamma_{32}^k$
Capacity Rollup	$\lvert x_{ij}^k \rvert \leq P_{ij}^k$	$\lvert x_{1,2}^k \rvert \leq P_{1,2}^k$ ... $\lvert x_{31,32}^k \rvert \leq P_{31,32}^k$
Inequality constraints	$\gamma_s^k \leq Y_s^k$	$\gamma_{15}^k \leq Y_{15}^k \ldots \gamma_{23}^k \leq Y_{23}^k$
Bounds	$\gamma_h^k = 0$	$\gamma_h^k = 0$
	$\sum_{(t)\in n}^{k} \gamma_t^k = Y_t^k$ $\gamma_t^k = Y_t^k$	$\gamma_9^k + \gamma_{11}^k = Y_{PM}^k$ $\gamma_{13}^k = Y_{13}^k$ $\gamma_{19}^k = Y_{19}^k$
	$P_{i,j}^k \geq 0$	$P_{1,2}^k \geq 0 \ldots . P_{31,32}^k \geq 0$
Operating Scenario	$P_{i,j}^{k,m} = 0$	$P_{1,2}^{k,m} = 0 \ldots P_{31,32}^{k,m} = 0$
Indices	$\delta(i,j) \in \{0,1\}$	$\delta(i,j) \in \{0,1\}$
Capturing aggregate solution	$P_{i,j} = \max_{(k,m)\in K,M} \left(P_{i,j}^{k,m}\right)$	$P_{1,2} = \max_{(k,m)\in K,M} \left(P_{1,2}^{k,m}\right) \ldots .$ $P_{31,32} = \max_{(k,m)\in K,M} \left(P_{31,32}^{k,m}\right)$

In this example, the SUBFLOW formulation adopts the M-1 survivability (Trapp, 2015) by looping a 144×176 Operational Matrix (see Table 16) as many as the quantity of arcs in the PPS network, i.e., 32 arcs. The objective function for the PPS study, which is given in Table 14, is in the first row and columns 1 to 108 in the Operational Matrix (see Table 16). To define variables α and β in (located in the first row and the first 72 columns) there were two assumed 'standard' edge components.

In this formulation, the network solution can be used into two different ways. The first one was termed as the 'binary variables' method that minimised the space taken by PPS connections using coefficients α and β. These coefficients categorised arcs in the PPS network to a certain standard edge component via binary decisions $\delta_{i,j}$. The second one was the 'integer variables' method, which also minimised the value of multiplication between the power to volume ratio $\lambda_{i,j}$ and the power $P_{i,j}$. The power to volume ratio $\lambda_{i,j}$ quantifies the power $P_{i,j}$ for each set of arcs connecting a node i and a node j into a discrete volume. By assuming some variables related to the PPS cabling specifications, the power to volume ratio $\lambda_{i,j}$ was obtained. Since there were unique x, y, z locations for each node from the DBB synthesis, the distance between nodes $L(i,j)$ could be calculated (see Table 15).

Table 15: Assumed variables in the PPS study

Variable	Description	Value
α	Binary coefficient of first category for cable sizing via the binary variables in the objective function	1.4 MW
β	Binary coefficient of second category for cable sizing via the binary variables in the objective function	4.8 MW
$\lambda_{i,j}$	Power to volume ratio for sizing via the integer variables in th objective function	$\frac{1.043 \times 10^{-5} m^2}{kW} L_{ij}$

In this case study, SUBFLOW did not just seek the minimum space for PPS cabling but also satisfied several constraints. These constraints were developed to show the distinctive SSK PPS operating conditions. In these constraints, k is an indexed scenario within a set of operating conditions K to represent various operating conditions, such as snorting and submerged conditions. In this PPS study, only the snorting (and transit) condition was considered, where the SEDs become the highest load in the PPS network, letting operating condition $k = 1$. The continuity formulation ensures the flow variable or flow path x entering and leaving a node n from a node i or j within a set of nodes n is equal to the amount of commodity γ at that node n and is preserved throughout the arcs A, except at relevant sources and targets. This equation is indicated in rows 2 to 33 and columns 109 to 176 in the Operational Matrix (Table 16).

For bidirectionality, the flow variable $x_{i,j}$ was 'rolled up' (Trapp, 2015) and converted to power capacity flow $P_{i,j}$ as the decision variables in SUBFLOW. Thus, the required power P_{ij}, as the decision variables in the SUBFLOW formulation, is always positive. This formulation is located in two parts in the Operational Matrix (see Table 16): rows 35 to 106 and columns 72 to 144; rows 143 to 144 and columns 109 to 144. The bounds in the formulation define the amount of power source and demand Y in the PPS network. The source nodes in the PPS study were the PGs, i.e., nodes 15, 17, 21, and 23 (see Table 16). This equation is assigned at rows 143 to 144 and columns 145 to 176 in the Operational Matrix framework. The bounds for hub nodes were set to zero. The examples of hub nodes in the PPS study were nodes 1, 2, 3, etc (see Figure 6). This equation is assigned at rows 143 to 144 and columns 145 to 176 in the Operational Matrix.

For applying the M-1 survivability by Trapp (2015) in this PPS network problem, each operating condition k is associated with an edge loss scenario m (the flow was set to zero) within a set of damaged scenarios M. This equation was applied by setting the upper bound of a power capacity flow $P_{i,j}$ to zero in the Operational Matrix, which is located at row 144 and columns 72 to 108. This setup forced the solver to be unable to use that arc and then search for an alternative set of flowpaths in the network.

Table 16: The Operational Matrix of the simplified PPS study, which is developed based on the formulation in Table 14

		Formulation for arcs				Formulation for nodes		
		36	36	36	36	32		
	No	**1-36**	**37-72**	**72-108**	**109-144**	**145-176**		**177**
Objective Function	**1**	$\boldsymbol{\alpha\,\delta_{i,j}}$	$\boldsymbol{\beta\,\delta_{i,j}}$	$\boldsymbol{\lambda_{i,j}\,P_{i,j}}$	$\sum_{(i,j)\in E} x_{i,n} - \sum_{(i,j)\in E} x_{n,j}$	$-\gamma_n$	=	0
	2	0	0	0	+1	+1		
	.	0 . . .	0 . . .	0 . . .	or . . .	or . . .		
Equality constraints matrix for continuity	.	0	0	0	-1	-1		0
	.	.	.	.	.	.		
		.	.	.	.	.		
	33	.	.	.	.	.		
	34					See LP Formulation for the PPS study		
				$-P_{i,j} \pm x_{i,j}$			≤	0
	35	0	0	-1	-1	0		
	.	0 . . .	0 . . .	-1 . . .	-1 . . .	0 . . .		
	.	0	0	-1	-1	0		
	.	.	.	.	.	.		
		.	.	.	.	.		0
	70	.	.	.	.	.		
	71	0	0	-1	+1	0		
	.	0 . . .	0 . . .	-1 . . .	+1 . . .	0 . . .		
Inequality constraints matrix for bidirectionality and binary variables	.	0	0	-1	+1	0		
	.							
		.	.	.	.	.		
		.	.	.	.	.		
	106	.	.	.	.	.		
		$\boldsymbol{\alpha\,\delta_{i,j}}$	$\boldsymbol{\alpha\,\delta_{i,j}}$	$P_{i,j}$			≤	0
	107	$\boldsymbol{-\alpha}$	$\boldsymbol{-\beta}$	**1**	0	0		
	.	$\boldsymbol{-\alpha}$ . . .	$\boldsymbol{-\beta}$ . . .	**1** . . .	0 . . .	0 . . .		
	.	$\boldsymbol{-\alpha}$	$\boldsymbol{-\beta}$	**1**	0	0		0
	.	.	.	.	.	.		
	.	.	.	.	.	.		
	142	.	.	.	.	.		
Lower bounds matrix	**143**	. . . 0 . . .	. . . 0 . . .	**. . . 0 . . .**	. . . −inf . . .	See LP Formulation for the PPS study		
Upper bounds matrix	**144**	. . . 1 . . .	. . . 1 . . .	**. . . inf . . .**	. . . inf . . .			
		$\boldsymbol{\delta}(i,j) \in \{0,1\}$		$\boldsymbol{P^k_{i,j} \geq 0}$	$\boldsymbol{\lvert x^k_{ij}\rvert \leq P^k_{ij}}$			

The δ in Equation in Table 16 serves as the binary decision to classify a capacity of an edge i to j to achieve certain standards for an edge component (type α and β). This equation is assigned at rows 143 to 144 and columns 1 to 72 in the Operational Matrix (see Table 16).

The redundant Propulsion Motors (PM)s were set as user nodes, but the solver could only select one PM to be online in an operating condition k. Other user nodes in the same operating condition (snorting) were set as the hard constraints. These are the batteries (SED) charging demands. Therefore, in the PPS study, the user nodes t were PMs and SEDs (nodes 9, 11, 13 and 19 in Table 14). These equations are shown in rows 34 and 143 to 144 and columns 145 to 176 in the Operational Matrix (see Table 16). Finally, the network solutions from the solver, which consisted of numerical data in a matrix, were presented in Table 17. This used the last equation in Table 14.

Using the Operational Matrix framework, the solver was able to find the solutions. Table 17 shows three different set of network solutions. The first solution is referred to as a "conservative" solution as it was obtained by selecting the maximum possible power flow in the PPS problem, i.e., maximum power available from the four PGs. The second and third solutions were based on the Objective Function of the SUBFLOW PPS explained above. The three solutions can be used as a basis for sizing the PPS cabling although it was recognised that the aft part of the PPS network would have required further operating scenarios k to be considered beyond snorting and transit (e.g., sprint condition). In this simple PPS example, there were found to be three possible options, the designer was able to choose between a smaller space solution (3 m^3 and 5 m^3) or the conservative solution (10 m^3) for the PPS cable sizing. Another example is provided and thus the next section outlines another version of SUBFLOW formulation for an SSK power load network at a high level.

3.3 Applying the Operational Matrix using high-level submarine power sizing variables

Network can also be used to represent the constants, coefficients, and variables within the concept design model as opposed to the items of the distributed ship service systems. This means the nodes are not directly representing the actual distributed ship service systems equipment unlike the previous examples. Therefore, a mathematical relationship for diesel power (P_{DIES}) sizing was used, which is given in Figure 7 (top) (Burcher and Rydill, 1994). This algorithm expresses that the power output that a diesel engine fit must be able to satisfy the electric service demand for charging batteries, propulsion load, and hotel load, as well as the likely inefficiencies and margins required to accomplish snorting operations.

$$P_{DIES} = \left[\overset{A}{(1+m)}\overset{B}{(I_{max} \times 2.4 \times N_c)} + \frac{\overset{C}{P_s'}}{\underset{E}{\eta_{el}}} + \overset{D}{H_{snort}}\right]\left(\frac{1}{\underset{F}{\eta_{conv}}}\right)$$

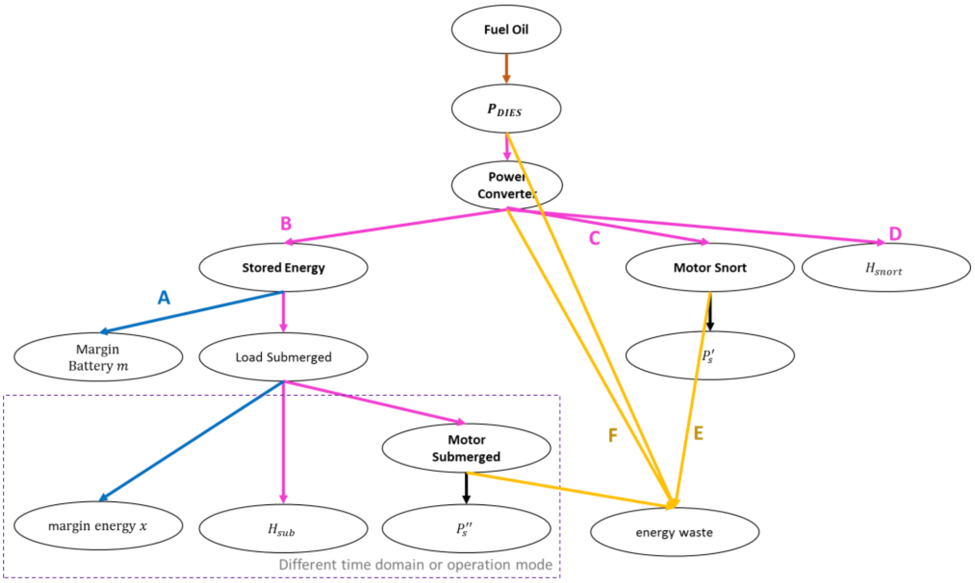


Figure 7: A high level SSK power system problem based on to the power sizing algorithm due to Burcher & Rydill (1994)

Table 17: Sizing results of the Power and Propulsion Systems (PPS) study

Arc No	**Node**		**Power to volume ratio $\lambda_{i,j}$ (m^3/kW)**	**Conservative Result**		**Integer Variables Result**		**Binary Variables Result**			
	i	j		Power $P_{i,j}$	Volume $V_{i,j}$	Power $P_{i,j}$	Volume $V_{i,j}$	Alpha α	Beta β	Power $P_{i,j}$	Volume $V_{i,j}$
				(kW)	(m^3)	(kW)	(m^3)			(kW)	(m^3)
1	1	2	5.52E-05	6291	0.347	0	0.000	-	-	0	0.000
2	1	25	8.68E-05	6291	0.546	0	0.000	-	-	0	0.000
3	2	3	5.84E-05	6291	0.367	346	0.020	yes	-	1400	0.082
4	2	10	4.31E-05	6291	0.271	346	0.015	yes	-	1400	0.060
5	3	4	4.02E-06	6291	0.025	1401	0.006	yes	-	1400	0.006
6	3	12	4.31E-05	6291	0.271	1401	0.060	yes	-	1400	0.060
7	4	5	7.72E-05	6291	0.486	3280	0.253	-	yes	4890	0.378
8	4	14	4.31E-05	6291	0.271	2934	0.127	-	yes	4890	0.211
9	5	6	7.30E-05	6291	0.459	1680	0.123	-	yes	4890	0.357
10	5	16	1.18E-05	6291	0.074	1600	0.019	-	yes	4890	0.058
11	6	7	1.71E-04	6291	1.076	3014	0.515	-	yes	4890	0.836
12	6	18	1.18E-05	6291	0.074	1600	0.019	-	yes	4890	0.058
13	7	8	1.05E-04	6291	0.660	0	0.000	-	-	0	0.147
14	7	20	4.31E-05	6291	0.271	3014	0.130	-	yes	4890	0.211
15	8	32	8.68E-05	6291	0.546	0	0.000	-	-	0	0.121
16	9	10	2.61E-06	6291	0.016	346	0.001	yes	-	1400	0.004
17	10	26	4.36E-05	6291	0.275	0	0.000	-	-	0	0.000
18	11	12	2.61E-06	6291	0.016	346	0.001	yes	-	1400	0.004
19	12	27	4.36E-05	6291	0.275	1401	0.061	yes	-	1400	0.061
20	13	14	2.74E-05	6291	0.172	2934	0.080	-	yes	4890	0.134
21	14	28	4.36E-05	6291	0.275	2934	0.128	-	yes	4890	0.213
22	15	16	2.61E-06	6291	0.016	1600	0.004	-	yes	4890	0.013
23	17	18	2.61E-06	6291	0.016	1600	0.004	-	yes	4890	0.013
24	19	20	2.74E-05	6291	0.172	2934	0.080	-	yes	4890	0.134
25	20	31	4.36E-05	6291	0.275	3014	0.132	-	yes	4890	0.213
26	21	22	2.61E-06	6291	0.016	1600	0.004	-	yes	4890	0.013
27	22	29	1.24E-05	6291	0.078	1600	0.020	-	yes	4890	0.060
28	23	24	2.61E-06	6291	0.016	1600	0.004	-	yes	4890	0.013
29	24	30	1.24E-05	6291	0.078	1600	0.020	-	yes	4890	0.060
30	25	26	5.52E-05	6291	0.347	0	0.000	-	-	0	0.000
31	26	27	5.84E-05	6291	0.367	0	0.000	-	-	0	0.000
32	27	28	4.02E-06	6291	0.025	1401	0.006	yes	-	1400	0.006
33	28	29	7.72E-05	6291	0.486	3280	0.253	-	yes	4890	0.378
34	29	30	7.30E-05	6291	0.459	1680	0.123	-	yes	4890	0.357
35	30	31	1.71E-04	6291	1.076	3014	0.515	-	yes	4890	0.836
36	31	32	1.05E-04	6291	0.660	0	0.000	-	-	0	0.147
Total Volume					**10.865**		**2.723**				**5.243**

The network in Figure 7 shows the hierarchal sources and sinks of a SSK power system with several nodes starting from the fuel (oil) tankage node as the source of energy followed by the diesel generator node (quantified by P_{DIES}), which converts the fuel is chemical energy (brown) to electric energy. The electrical energy is then converted and distributed by a power converter node to the three main electric loads which are coloured in magenta: the energy storage or battery charging for fully submerged operation (B); the hotel load in the snorting operation (D); and the propulsion load in snorting operations (C). Further nodes have been modelled to represent margins (coloured in blue) for battery charging (A) and submerged energy (B), as well as efficiencies (F and E) coloured in yellow, which contribute to energy waste or power loss. The detailed heat due to battery charging and hotel load in the snorting operation was not considered in this modelling.

The properties of the nodes shown in Figure 7 are given in Table 18. The source node in this study was the Fuel Oil (FO) node, while the rest of the terminal nodes were sinks and between the terminal nodes, there were hub nodes. Unlike terminal nodes, hub nodes have to have at least one in-degree and one out-degree flow. The hub nodes shown in Table 18 are the Diesel Generator (DG), the Power Converter (PC), the Stored Energy (SE), the Margin Battery (MM), the Load Submerged (LS), Motor Submerged (MS), and Motor Snort (MT). Compared to the AFO approach (Brown, 2020), each arc in the SUBFLOW network also focuses on one commodity, which is energy (chemical, electrics, mechanical, or heat loss). However, in the AFO approach, there could be a non-energy flow, such as data flow (carrying binary 0 and 1 numerical data), as the 'parallel' commodity in the AFO formulation (Robinson, 2018). Reducing the number of commodities within the SUBFLOW then reduced the number of inputs and the complication in the network formulation, making the SUBFLOW more appropriate to be applied early in the design process, as in the implementation shown in Table 18. The energy storage was also explicitly modelled as the Load Submerged (LS) node in this network.

Table 18: Nodes properties for an SSK power system network in Figure 7

Node Name	Relevant Variable	Node Identification	SUBFLOW Setup	Node Type	Data
Fuel Oil	P_{Fuel}	FO	$P_{FO} \geq 0$	Terminal	(Output)
Diesel Generator	P_{Dies}	DG	$P_{DG} \geq 0$	Hub	(Output)
Power Converter	P_{Conv}	PC	$P_{PC} \geq 0$	Hub	(Output)
Stored Energy	P_{Batt}	SE	$P_{SE} \geq 0$	Hub	(Output)
Margin Battery	m	MM	$P_{MM} \geq 0$	Hub	(Output)
Load Submerged	-	LS	$P_{LS} \geq 0$	Hub	(Output)
Margin Energy	x	MX	$P_{MX} \geq 0$	Terminal	(Output)
Hotel Submerged	H_{sub}	HS	$\boldsymbol{P_{HS} = H_{sub}}$	Terminal	280 kW
Motor Submerged	P''_{Motor}	MS	$P_{MS} \geq 0$	Hub	(Output)
Velocity Submerged	P''_s	VS	$\boldsymbol{P_{VS} = P''_s}$	Terminal	68 kW
Motor Snort	P'_{Motor}	MT	$P_{MT} \geq 0$	Hub	(Output)
Velocity Snort	P'_s	VT	$P_{VT} = P'_s$	Terminal	(Output)
Hotel Snort	H_{snort}	HT	$P_{HT} = H_{snort}$	Terminal	224 kW
Heat Loss	-	HE	$P_{HE} \geq 0$	Terminal	(Output)

In this example, a formulation used in the AFO approach (Robinson, 2018) was applied to continuity constraints and to define how much energy could come in and out of a hub node, denoted as an energy coefficient e_i in this SUBFLOW simulation (see Table 19 and the continuity in Table 21). For example, at the Diesel Generator (DG) node, 100% of the incoming energy flow from the Fuel Oil (FO) node would be converted to the Power Converter (PC) node as the electric energy (48%) and Heat Loss (HE) node (52%). This split could be said to be similar to the Sankey diagram that can be used to breaking down energy inputs and outputs (Kennedy and Sankey, 1898). Thus, all hub nodes' energy coefficients e_i in this SUBFLOW network (Figure 7) are provided in Table 19.

Table 19: Arcs properties for an SSK power system network in Figure 7

Arc $(\boldsymbol{i},\boldsymbol{j})$	Energy	Colour Code	SUBFLOW Setup $\sum_{(i,j)\in E} P_{i,j} - e_i P_i = 0$
(FO,DG)	Chemical	Brown	$P_{FO,DG} = P_{FO}$
(DG,PC)	Electrical	Magenta	$P_{DG,PC} = 48\%\ P_{DG}$
(DG,HE)	Heat	Yellow	$P_{DG,HE} = 52\%\ P_{DG}$
(PC,SE)	Electrical	Magenta	$P_{PC,SE} = 98\%\ P_{PC} - P_{PC,MT} - P_{PC,HT}$
(PC,MT)	Electrical	Magenta	$P_{PC,MT} = 98\%\ P_{PC} - P_{PC,SE} - P_{PC,HT}$
(PC,HT)	Electrical	Magenta	$P_{PC,HT} = 98\%\ P_{PC} - P_{PC,SE} - P_{PC,MT}$
(PC,HE)	Heat	Yellow	$P_{PC,HE} = 2\%\ P_{DG}$
(SE,MM)	Electrical	Blue	$P_{SE,MM} = 4.8\%\ P_{SE}$
(SE,LS)	Electrical	Magenta	$P_{SE,LS} = 95.2\%\ P_{SE}$
(MT,HE)	Heat	Yellow	$P_{MT,HE} = 3\%\ P_{MT}$
(MT,VT)	Mechanical	Black	$P_{MT,VT} = 97\%\ P_{MT}$
(LS,MS)	Electrical	Magenta	$P_{LS.MS} = 64\%\ P_{LS} - P_{LS.HS}$
(LS,HS)	Electrical	Magenta	$P_{LS,HS} = 64\%\ P_{LS} - P_{LS.MS}$
(LS,MX)	Electrical	Blue	$P_{LS,MX} = 36\%\ P_{LS}$
(MS,HE)	Heat	Yellow	$P_{MS,HE} = 3\%\ P_{MS}$
(MS,VS)	Mechanical	Black	$P_{MS,VS} = 97\%\ P_{MS}$

All arcs in the network were not capped and thus it can be any positive values $0 \leq P_{i,j} \leq \infty$ (or Inf). This will also be the case for the supply node, the Fuel Oil (FO) γ_s, all hub nodes γ_h, and some target nodes γ_t, such as the Margin Energy (MX), the Motor Submerged (MS), the Motor Snort (MT), and the Heat Loss (HE) (see the bounds in Table 14. The capacities of the user nodes γ_t in the network then need to be calculated, which are dependent on the operating scenarios (see Table 14). They are the Hotel Submerged (HS), the Velocity Submerged (VS), the Velocity Snort (VT), and the Hotel Snort (HT). Finally, the coefficients of the objective function coefficient $c_{i,j}$ in this SUBFLOW example were set to zero (see Objective Function equation in Table 14), because the aim of this optimisation example was not to cost the distributed ship service systems configuration, as in the case of Trapp's (2015) NSMCF investigation or Robinson's (2018) AFO study (including its variants (Parsons et al., 2020)). In this study, SUBFLOW was used to solve the energy balance, through a linear programming, set of equations. This ensured that the total energy demand on the submarine would be equal to the total energy available, indicating an initial systems design balance. Thus, in this SUBFLOW example, the network styles were proposed on the basis of prior expert knowledge and were deliberately not validated by analysis in early stage of ship design. Nonetheless, the SUBFLOW network created could have provided a suitable basis for further analyses in subsequent design phases, if required.

Table 20: Linear programming formulation and realisation of an SSK power system network in Figure 7

Linear Programming Formulation	Mathematical Notation	Realisation
Objective Function:	$min. \sum_{(i,j)\in a} c_{i,j} P_{i,j}$ where $c_{i,j} = 0$	$c_{FO,DG}P_{FO,DG} + \ldots + c_{MS,VS}P_{MS,VS}$

Subject To		
Linear Programming Formulation	**Mathematical Notation**	**Realisation**
Continuity	$\sum_{(i,j)\in E} P_{i,j} - e_i P_i = 0$	$P_{FO,DG} - e_{FO} P_{FO} = 0 \ldots$ $P_{DG,HE} + P_{PC,HE} + P_{MT,HE} + P_{MS,HE} - e_{HE} P_{HE} = 0$
Bounds	$0 \leq P_{i,j} \leq \infty$ (or Inf)	$0 \leq P_{FO,DG} \leq \infty \ldots$ $0 \leq P_{MS,VS} \leq \infty$
	$0 \leq \gamma_s \leq \infty$ (or Inf)	$0 \leq P_{FO} \leq \infty$
	$0 \leq \gamma_h \leq \infty$ (or Inf)	$0 \leq P_{DG} \leq \infty, 0 \leq P_{PC} \leq \infty, 0 \leq P_{SE} \leq \infty,$ $0 \leq P_{MM} \leq \infty\,, 0 \leq P_{LS} \leq \infty, 0 \leq P_{MS} \leq \infty,$ $0 \leq P_{MT} \leq \infty$
	$0 \leq \gamma_t \leq \infty$ (or Inf)	$0 \leq P_{MX} \leq \infty$ *and* $0 \leq P_{HE} \leq \infty$
Investigated Operating Scenarios	$\gamma_t = Y_t$	$\gamma_{HT} = Y_{HT}, \gamma_{VT} = Y_{VT}, \gamma_{VS} = Y_{VS}, \gamma_{HS} = Y_{HS}$

The Operational Matrix for this particular SUBFLOW example is outlined in Table 21. Since undirected network or bidirectional network contains more information than an undirected network (Mukti, 2022), the size of the Operational Matrix would have become quite large if bidirectionality had had to be considered (i.e., a $[61 \times 47]$ matrix). Nonetheless, in this case, bidirectionality was not necessary as there had to be no backward flow from the target to the source nodes and thus the Operational Matrix has only 29 rows and 31 columns. Compared to the Operational Matrix for Sections 3.1 and 3.2, the Operational Matrix in this example shows how to arrange the energy coefficient from the AFO approach (Parsons et al., 2020) in the matrix, which is reflected as the coefficient e_i for the continuity constraints (rows 2 to 27 and columns 17 to 30).

By using the Operational Matrix in Table 21, the solver can provide the network solution as shown in Figure 8. Figure 8 shows 5.7 MW of power is transferred from the Fuel Oil node to the Diesel Generator node. The Diesel Generator node then converted the 5.7 MW of the fuel flow to 2.7 MW to Power Converter node (as electrical flow shown in magenta) and 2.9 MW to Heat Loss node (as waste heat shown in yellow). At the Power Converter node, the 2.7 MW of electrical flow was divided into 2.3 MW electrical flow for Stored Energy node, 166 kW for Motor Snort node, 224 kW to Hotel Snort node, and 55 kW to Heat Loss node. For the Stored Energy various flows simulate how much energy is needed during the submerged operating condition, i.e., a different time domain from the snorting operating condition. All of the flows shown in Figure 8 satisfied the SUBFLOW constraints given in Table 21.

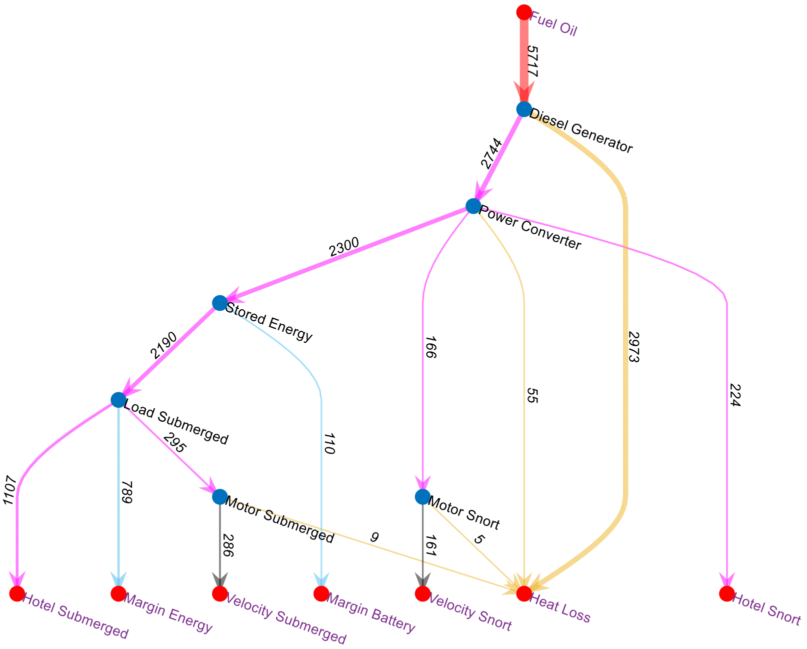


Figure 8: SUBFLOW solution for an SSK power system

Table 21: The Operational Matrix of a high level SSK power system in Figure 7, which is developed based on the formulation in Table 20.

	No	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30		31
Objective Function	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
		Arcs																Nodes															
									$\sum_{(i,j)\in E} P_{i,j}$															$-e_i P_i$								=	0
		$P_{FO,DG}$	$P_{DG,PC}$	$P_{DG,HE}$	$P_{PC,SE}$	$P_{PC,MT}$	$P_{PC,HT}$	$P_{PC,HE}$	$P_{SE,MM}$	$P_{SE,LS}$	$P_{MT,HE}$	$P_{MT,VT}$	$P_{LS,MS}$	$P_{LS,HS}$	$P_{LS,MX}$	$P_{MS,HE}$	$P_{MS,VS}$	P_{FO}	P_{DG}	P_{PC}	P_{SE}	P_{MM}	P_{MT}	P_{HT}	P_{HE}	P_{VT}	P_{LS}	P_{MS}	P_{VS}	P_{HS}	P_{MX}		
Equality constraints matrix for continuity	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0		0
	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0		0
	4	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-0.48	0	0	0	0	0	0	0	0	0	0	0	0		0
	5	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-0.52	0	0	0	0	0	0	0	0	0	0	0	0		0
	6	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0		0
	7	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	-0.98	0	0	0	0	0	0	0	0	0	0	0		0
	8	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	-0.02	0	0	0	0	0	0	0	0	0	0	0		0
	9	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0		0
	10	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	-0.952	0	0	0	0	0	0	0	0	0	0		0
	11	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	-0.048	0	0	0	0	0	0	0	0	0	0		0
	12	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0		0
	13	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0		0
	14	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	-0.64	0	0	0	0		0
	15	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	-0.36	0	0	0	0		0
	16	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1		0
	17	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0		0
	18	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0		0
	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	-0.970	0	0	0		0
	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	-0.030	0	0	0		0
	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	-1	0	0		0
	22	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0		0
	23	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	-0.970	0	0	0	0	0	0	0	0		0
	24	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	-0.030	0	0	0	0	0	0	0	0		0
	25	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0		0
	26	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0		0
	27	0	0	1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0		0
Lower bounds matrix	28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**224**	0	**160**	0	0	**285**	**1107**	0		
Upper bounds matrix	29	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	**224**	Inf	**160**	Inf	Inf	**285**	**1107**	Inf		
									$0 \le P_{i,j} \le \infty$															Investigated operating conditions/scenarios (see Table 20)									

4. THE OPERATIONAL MATRIX FRAMEWORK APPLIED TO THE UCL NETWORK BLOCK APPROACH

Section 3 addresses how the Operational Matrix Framework can be used to assess different types of network flow formulations. This section presents the setup of the Operational Matrix Framework that has been adopted in the UCL Network Block Approach (Mukti, 2022). Unlike the previous examples, the SUBFLOW formulation in the UCL Network Block Approach was devised to be ship design efficient, yet without losing the advantages of capturing the complexity of distributed ship service systems using a range of applicable network tools. In the UCL Network Block Approach, spreadsheet-based tools are used to define the ship design and its distributed ship service systems (Mukti et al., 2022). The tools specific for defining the network configuration and SUBFLOW inputs for the distributed ship service systems are the Component Granularity Program (CGP) and System Connection Program (SCP) (see (Mukti et al., 2022)).

Table 22 shows the example of the CGP inputs: the type of components (nodes), which were either terminal or hub nodes; the equipment load demand or maximum capacity was used to define the lower and upper bounds for the SUBFLOW in various operating conditions, for example, snort or sprint submerged); the objective function coefficient, which was set to zero; the energy coefficients *e* of each component up to 15 different types of distributed ship service systems commodities; and the logical layout (x, y, z coordinates) to create a "logical" multiplex network.

The energy coefficients *e* of each component node in Table 22 are defined as follows:

- The energy that enters a node is expelled 100% outside the node (IN=-1). This option was used for terminal source nodes, such as fuel, or terminal sink nodes, such as propulsion load.

- The energy that enters a node is dispersed to different types of energy in a form of some fraction (IN=fractional OUT). This reflects the Sankey Diagram practice and could have been used for electrical consumer nodes, including energy storage.

- The energy that enters a node is determined by the proportion of the energy from at least two different nodes in different systems. This option could have been used for modelling the fuel-air (energy) ratio of the diesel generator.

- The energy that might have entered a node could have been specified as a fraction of the total heat received at the node and that fraction of energy that has not been forwarded beyond that node (fractional IN=OUT). This choice could be used to describe the 'coefficient of performance' of cooling systems components.

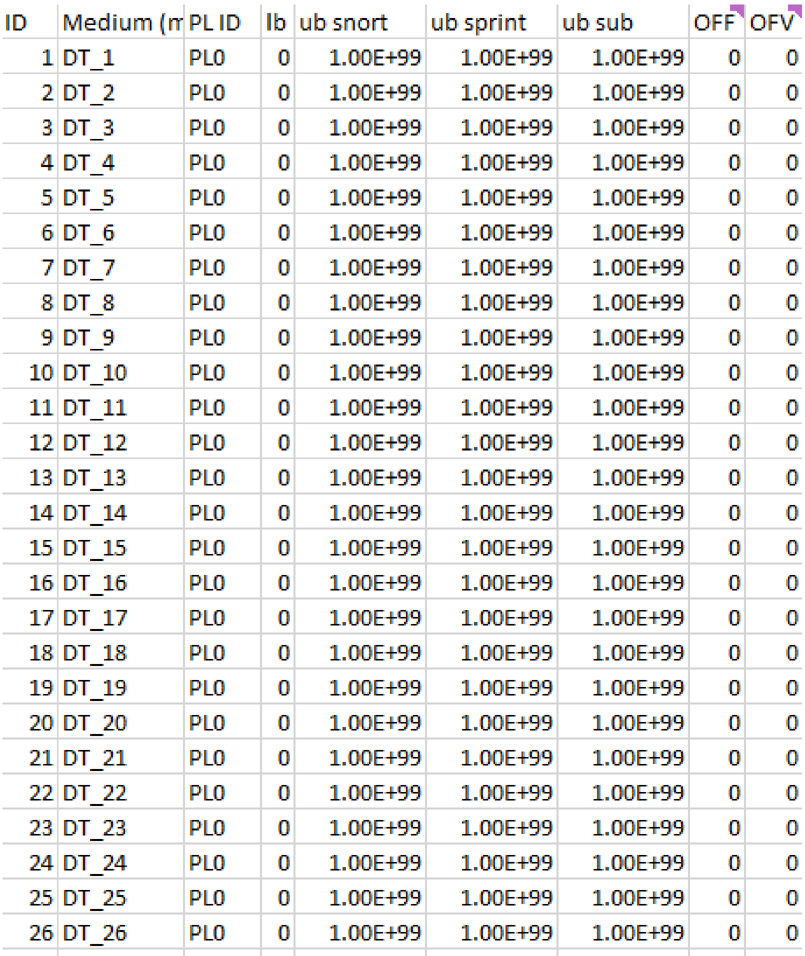
- A 'child' node could receive 100% energy from two parent nodes from different systems and then store 100% energy output to that child node. This could have been used to model sink nodes on the vessel, for example, a seawater node.

Table 22: Example of the CGP inputs for performing SUBFLOW

Node	Name	Description	Terminal/Hub	Energy (kW)	FO		EL		ME		HVIN		HVHE		HVEX		LO		CW		FW/SW		SW HE	
			Type	lb	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT
1	BB_DB_DT_CO_SS_f	distribution fwd	terminal	10.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
2	BB_DB_DT_CO_SS_m	distribution mid	terminal	10.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
3	BB_DB_DT_CO_SS_a	distribution aft	terminal	10.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
4	BB_DB_DT_MA_CO	main command CCS 80 etc	terminal	20.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
5	BB_DB_FO_SE_TK_p	settling tank port	terminal	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
6	BB_DB_FO_SE_TK_s	settling tank stbd	terminal	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
7	BB_DB_FO_CE_PU_p	centrifugal purifier port	hub	0.000	OUT	-1.000	0.010	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
8	BB_DB_FO_CE_PU_s	centrifugal purifier stbd	hub	0.000	OUT	-1.000	0.010	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
9	**BB_DB_EL_GT_EN_p**	Gas turbine generator set	hub	0.000	0.064	0.000	-	-0.480	-	0.000	0.936	0.000	-	0.000	-	-0.439	-	-0.081	-	0.000	-	0.000	-	0.000
16	**BB_DB_EL_GT_EN_s**	Gas turbine generator set	hub	0.000	0.064	0.000	-	-0.480	-	0.000	0.936	0.000	-	0.000	-	-0.439	-	-0.081	-	0.000	-	0.000	-	0.000
10	BB_DB_EL_PD_AL	power distribution aft load	hub	0.000	-	0.000	IN	-0.900	-	0.000	-	0.000	-	-0.100	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
11	BB_DB_EL_PD_ML	power distribution mid load	hub	0.000	-	0.000	IN	-0.900	-	0.000	-	0.000	-	-0.100	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
12	BB_DB_EL_PD_FL	power distribution forward load	hub	0.000	-	0.000	IN	-0.900	-	0.000	-	0.000	-	-0.100	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
13	BB_DB_ME_PM_MO_p	propulsion module port	hub	0.000	-	0.000	IN	0.000	-	-0.900	-	0.000	-	-0.100	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
14	BB_DB_ME_TT_BE_p	thrust block port	hub	0.000	-	0.000	-	0.000	IN	-0.900	-	0.000	-	0.000	-	0.000	-	-0.001	-	0.000	-	0.000	-	0.000
15	BB_DB_ME_PR_NE_p	propeller port	terminal	18500.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
17	BB_DB_ME_PM_MO_s	propulsion module stbd	hub	0.000	-	0.000	IN	0.000	-	-0.900	-	0.000	-	-0.100	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
18	BB_DB_ME_TT_BE_s	thrust block stbd	hub	0.000	-	0.000	-	0.000	IN	-0.900	-	0.000	-	0.000	-	0.000	-	-0.001	-	0.000	-	0.000	-	0.000
19	BB_DB_ME_PR_NE_s	proeller stbd	terminal	18500.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
20	BB_DB_HV_IN_BP_p	inlett bypass doors port	terminal	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
21	BB_DB_HV_IN_SI_p	intake silencing splitters port	hub	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
22	BB_DB_HV_IN_BP_s	inlett bypass doors stbd	terminal	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
23	BB_DB_HV_IN_SI_s	intake silencing splitters stbd	hub	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
24	BB_DB_HV_HE_ZA	heat zone aft	hub	0.000	-	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
25	BB_DB_HV_HE_ZM	heat zone mid	hub	0.000	-	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
26	BB_DB_HV_HE_ZF	heat zone fwd	hub	0.000	-	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000
27	BB_DB_HV_HE_AT_a	ATU aft	hub	0.000	-	0.000	0.020	0.000	-	0.000	-	0.000	OUT	0.000	-	0.000	-	0.000	-	-1.000	-	0.000	-	0.000
28	BB_DB_HV_HE_AT_m	ATU mid	hub	0.000	-	0.000	0.020	0.000	-	0.000	-	0.000	OUT	0.000	-	0.000	-	0.000	-	-1.000	-	0.000	-	0.000
29	BB_DB_HV_HE_AT_f	ATU forward	hub	0.000	-	0.000	0.020	0.000	-	0.000	-	0.000	OUT	0.000	-	0.000	-	0.000	-	-1.000	-	0.000	-	0.000
31	BB_DB_HV_EX_EJ_p	exhasust por	terminal	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000
30	BB_DB_HV_EX_SI_p	silincing splitter port	hub	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000
33	BB_DB_HV_EX_EJ_s	exhaust stbd	terminal	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000
32	BB_DB_HV_EX_SI_s	silincing splitter stbd	hub	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000	-	0.000	-	0.000	-	0.000	-	0.000
34	BB_DB_LO_HX_PT	LO HX	hub	0.000	-	0.000	0.070	0.000	-	0.000	-	0.000	-	0.000	-	0.000	OUT	0.000	-	0.000	-	-1.000	-	0.000
35	BB_DB_CW_HX_PT	CW HX	hub	0.000	-	0.000	0.061	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	OUT	0.000	-	-1.000	-	0.000
36	BB_DB_FW_SW_HX	FW/SW HX	hub	0.000	-	0.000	0.006	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	OUT	0.000	-	-1.000
37	BB_DB_SW_EX_EN	sea chest	terminal	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	-	0.000	IN	-1.000

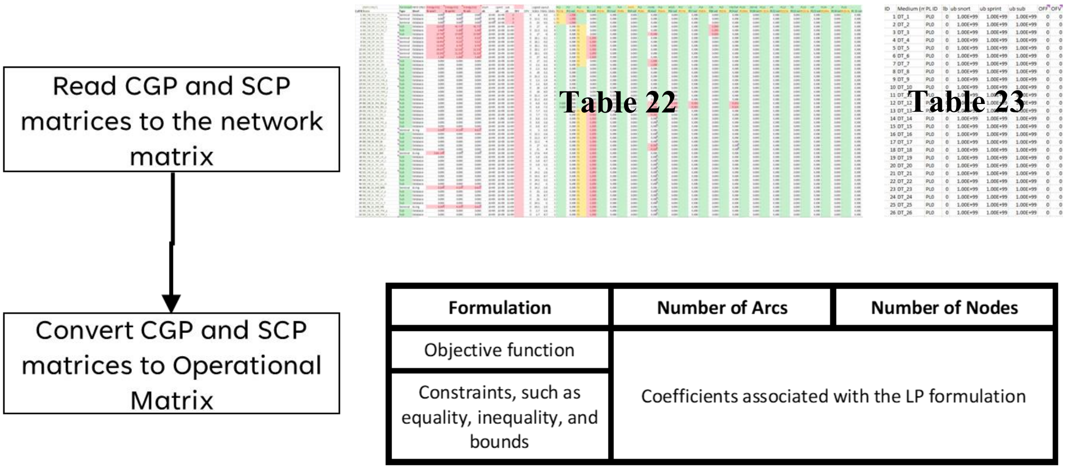
Like the CGP, the System Connection Program (SCP) also provides necessary inputs for the SUBFLOW formulation, particularly for the connections/arcs. As shown in Table 23, the input consists of the identification of distributed ship service systems commodity/ technology (e.g., DT for data, EL for electrics), the minimum and maximum capacity of the connection based on a given scenario (e.g., snort or sprint submerged for submarine case), and the objective function coefficient, which was also set to zero.

Table 23: Example of the SCP inputs for performing SUB/RFLOW



ID	Medium (m	PL ID	lb	ub snort	ub sprint	ub sub	OFF	OFV
1	DT_1	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
2	DT_2	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
3	DT_3	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
4	DT_4	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
5	DT_5	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
6	DT_6	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
7	DT_7	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
8	DT_8	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
9	DT_9	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
10	DT_10	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
11	DT_11	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
12	DT_12	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
13	DT_13	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
14	DT_14	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
15	DT_15	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
16	DT_16	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
17	DT_17	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
18	DT_18	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
19	DT_19	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
20	DT_20	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
21	DT_21	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
22	DT_22	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
23	DT_23	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
24	DT_24	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
25	DT_25	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0
26	DT_26	PL0	0	1.00E+99	1.00E+99	1.00E+99	0	0

Once these inputs had been defined, the inputs above (captured in CGP and SCP) needed to be converted into an Operational Matrix format so that the solver in MATLAB could produce the SUBFLOW network solution. The generation of the Operational Matrix format can be demanding in the early stage of ship design if it was not automated as there can be thousands of rows and columns for defining a network of distributed ship service systems SUBFLOW problem. Thus, to make SUBFLOW as efficient as possible, the generation of the Operational Matrix need to be automated (see Figure 9).


Read CGP and SCP matrices to the network matrix
Table 22
Table 23
Convert CGP and SCP matrices to Operational Matrix

Formulation	Number of Arcs	Number of Nodes
Objective function	Coefficients associated with the LP formulation	
Constraints, such as equality, inequality, and bounds		

Figure 9: Automatic generation of the Operational Matrix from CGP and SCP inputs

To make an automated generation of the Operational Matrix possible, a MATLAB script was developed. Figure 10 shows the process including reading the CGP and SCP inputs and storing them into the network matrix in MATLAB. The SUBFLOW formulation is next converted into the Operational Matrix format and the Operational Matrix is fed into a solver in MATLAB to find the SUBFLOW solution. If the solver fails to find a set of feasible solutions, the CGP and SCP inputs need to be evaluated and altered. Once the SUBFLOW solution is found, the data is stored back in the network matrix in MATLAB.

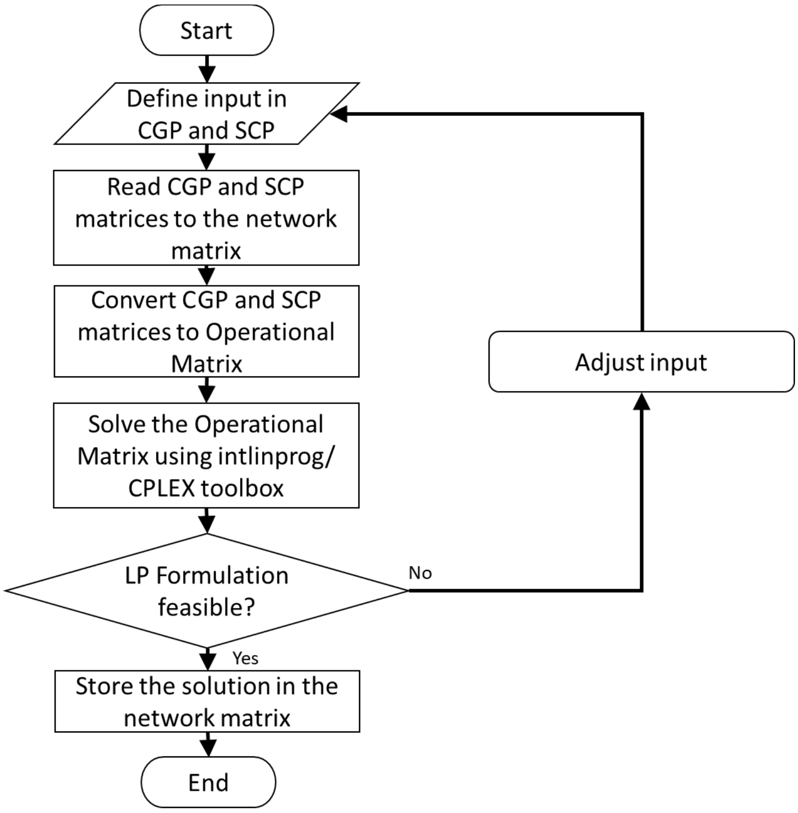

Start
Define input in CGP and SCP
Read CGP and SCP matrices to the network matrix
Convert CGP and SCP matrices to Operational Matrix
Solve the Operational Matrix using intlinprog/ CPLEX toolbox
LP Formulation feasible?
No
Adjust input
Yes
Store the solution in the network matrix
End

Figure 10: The procedure for automatic Operational Matrix generation

The pattern of the Operational Matrix for the UCL Network Block Approach is given in Table 24. The number of columns of the Operational Matrix depends on the number of arcs in the SUBFLOW network. This matrix is divided into several groups of boxes in rows. The first box of the matrix in Table 24 gives the objective function coefficients, which is set to zero to obtain the energy balance. The second box is allocated to continuity as well as the energy coefficient e for each node, which can be positive or negative. The third box contains the inequality constraints and the last two boxes consists of the lower bounds and the upper bounds. The lower and upper bounds are where the 'operational' aspect is defined, i.e., the supply or demand of a commodity of a node in each operating condition. Therefore, a different operating condition (e.g., snort and sprint submerged) requires a different Operational Matrix, which can be treated as a loop in MATLAB (see Figure 11).

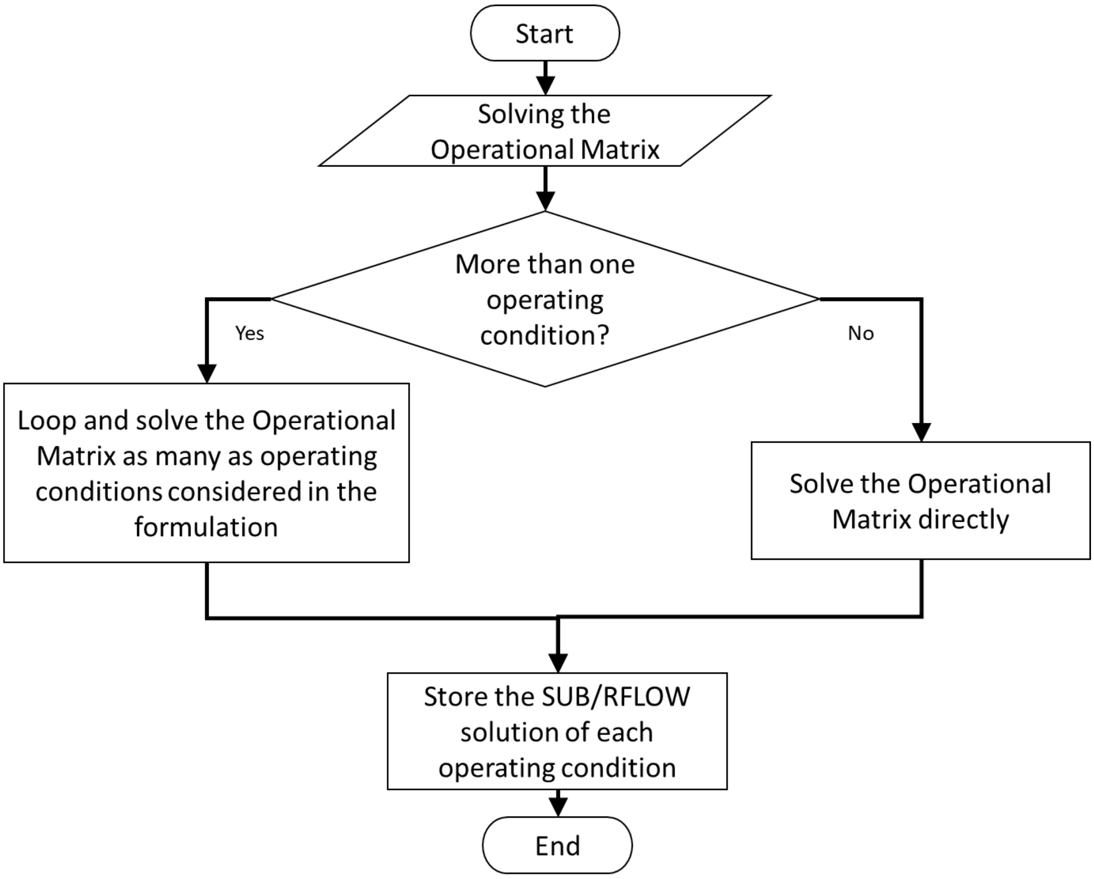

Start
Solving the Operational Matrix
More than one operating condition?
Yes
No
Loop and solve the Operational Matrix as many as operating conditions considered in the formulation
Solve the Operational Matrix directly
Store the SUB/RFLOW solution of each operating condition
End

Figure 11: Solving the Operational Matrix based on operating condition(s)

Compared to example Operational Matrices in the previous section, the matrix in Table 24, is scalable and simpler. This was to reduce the extent of the SUBFLOW analysis to be commensurate with early stage ship design. SUBFLOW was used to perform a steady state simulation of power flow in a distributed ship service systems network. SUBFLOW was employed to provide early estimates of the distributed ship service systems space and weight input as well as to explore distributed ship service systems options as part of the Requirement Elucidation process (Andrews, 2018).

The continuity constraints in the SUBFLOW network for the UCL Network Block Approach were hardcoded, i.e., automatically generated. However, the rest of the mathematical model for SUBFLOW formulation could be adjusted/ defined in the Component Granularity Program (CGP) and the System Connection Program (SCP). The formulation process was iterative and substantial to achieve a feasible network solution, i.e., if the formulation is incorrect, the solver will not be able to find the linear programming network solution (see Figure 10). Still, the SUBFLOW required more engineering and inputs than parametric approach, such as the configuration of the distributed ship service systems, also specifying its properties, and creating mathematical models for the energy balance analysis.

The SUBFLOW in the UCL Network Block Approach allows the style (Andrews, 2018) of distributed ship systems to be captured and can aid the designer to understand how a given system functions. Most importantly, when combined with the whole ship UCL Design Building Block approach (Andrews and Pawling, 2003), it enabled a more realistic distributed ship service systems synthesis to be undertaken. This was thus not just numeric but also addressed spatial/architectural aspects which a parametric approach lacking. However, it is not as detailed as collaborative analysis tools which are more appropriate to detailed design of such distributed ship service systems. Figure 12 shows an example of modelling a range of submarine systems: fuel (FO); electrical (EL); data (DT); mechanical (ME); hydraulics (HY); trim and ballast (TB); saltwater (SW); high-pressure air (HP); low-pressure air (LP); ventilation (HVIN/HE/EX); chilled water (CW); lubricating oil (LO); fresh water (FW) cooling systems (see (Mukti, 2022) for details).

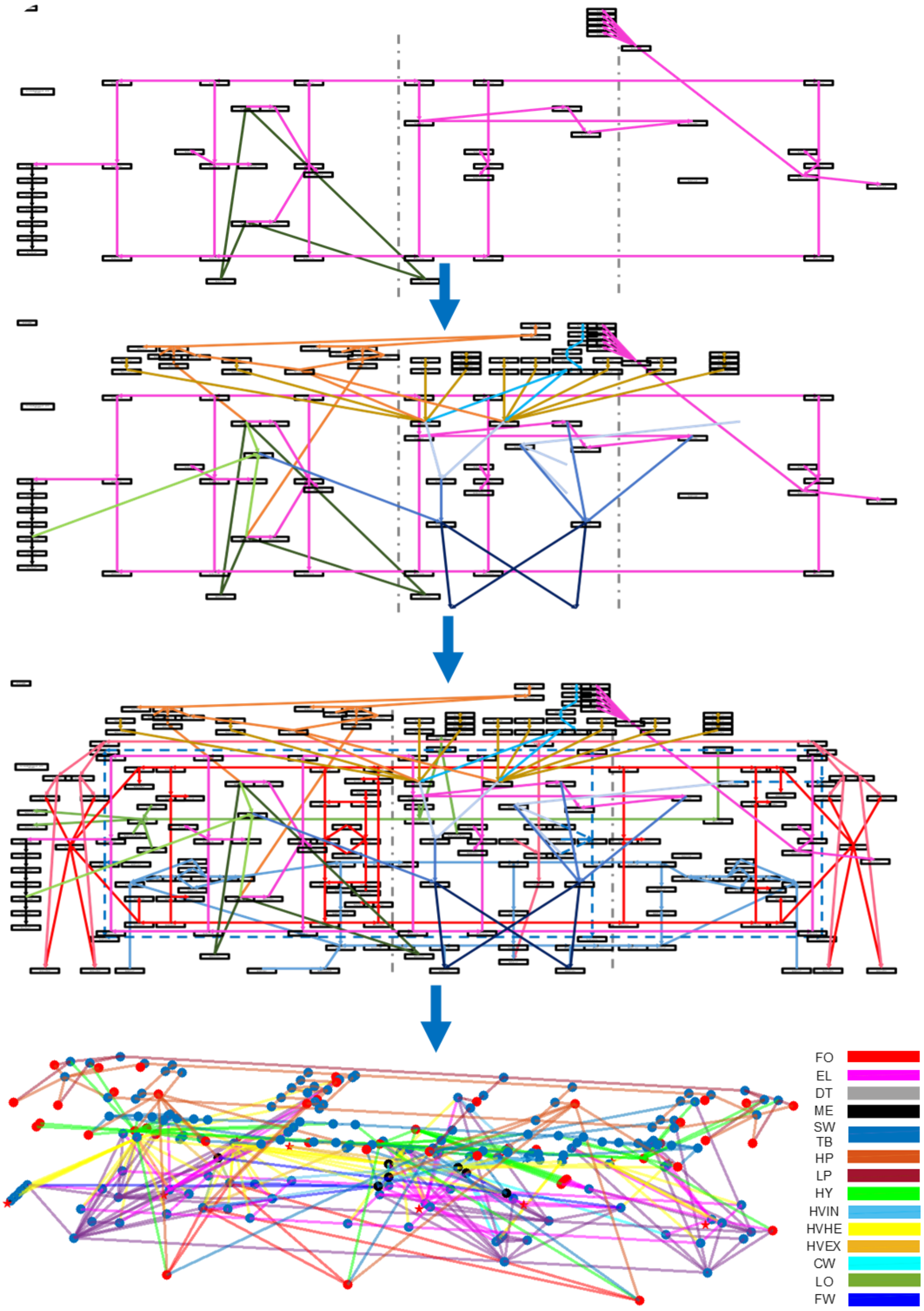


Figure 12: An example of SUBFLOW network development and solution in the UCL Network Block Approach (Mukti, 2022), showing energy flows in various distributed ship service systems

Table 24: The pseudo-Operational Matrix of SUB/RFLOW in the UCL Network Block Approach

	1, 2, … number of arcs	1, 2, … number of arcs	1, 2, … number of arcs	1
Objective Function	0 . . .	0 . . .	0 . . .	0
Equality constraints matrix for continuity	0 0 . . . 0 . . .	$+e$ or . . . $-e$ . . .	$+e$ or . . . $-e$ . . .	0
Inequality constraints matrix for bidirectionality	-1 -1 . . . -1 . . . -1 -1 . . . -1 . . .	-1 -1 . . . -1 . . +1 +1 . . . +1 . . .	0 0 . . . 0 . . 0 0 . . . 0 . . .	0
Lower bounds matrix	. . . 0 . . .	. . . -inf . . .	Based on input provided in the Component Granularity Program (CGP) and the System Connection Program (SCP)	
Upper bounds matrix	. . . inf . . .	. . . inf . . .		

5. CONCLUSION AND FUTURE WORK

There are several possible applications of the Operational Matrix Framework, such that it could be employed to suit several specific network flow formulations. Table 25 shows the high-level comparison between various Network Flow Optimisation setups. Generally, the SUB/RFLOW excludes the survivability analysis to allow enhancement in several aspects, such as the incorporation of a colour-coded 3D multiplex network with the labels showing how much energy flowing from system to system (Figure 12). The Operational Matrix Framework approach has enabled the solvers to be very efficient compared to millions of lines of CPLEX scripts (Brown, 2020). Most importantly, SUB/RFLOW enabled the incorporation of the 3D rich architecturally centred approach, the UCL DBB approach, which shows the potential benefits in assessing wholeship impact of a new (style) of distributed ship service systems design (Mukti, 2022). However, as aforementioned, the design data in the UCL Network Block Approach could have provided a suitable basis for further (survivability) analyses in subsequent design phases, if required.

Table 25: General comparison between different types of Network Flow Optimisation setup for naval ships applications

Formulation	**Network Flow Optimisation for Ship Systems Application**		
	NSMCF (Trapp, 2015)	**AFO** (Brown, 2020)	**SUBFLOW** (Mukti, 2022)
Objective Function	Procurement and installation cost	Procurement and installation cost	None (“constraints only” approach to find energy balance)
Solver	CPLEX	CPLEX	CPLEX toolbox (or *linprog*) in MATLAB
Software interface/ Programming language	MATLAB script to CPLEX script	MATLAB script to CPLEX script	Fully in MATLAB using Operational Matrix Framework
Survivability analysis	Yes	Yes	No
Can be used for sizing distributed systems	No	Yes	Yes
Application	Integrated Engineering Propulsion (IEP) plant	Surface ship systems	Submarine/ Surface ship systems (Mukti et al., 2024)
Using energy coefficient	No	Yes	Yes
Using flow capacity	Yes	Yes	No
Hot and cool model	Yes	Yes	No/Simplified
Multilayer network/ all arcs visible to inspect the energy flow between systems	No	No	Yes, using SUB/RFLOW Multiplex Framework
Support 3D rich ship architecture definition (e.g., equipment arrangement and routings)	No	No	Yes, via Paramarine-SURFCON

In this paper, studies range from the simplest application of Operational Matrix Framework to an example of one of the more complex Operational Matrix applications to the 3D multiplex submarine systems problem in the last section. The use of the proposed Operational Matrix Framework can reveal the relationship between objective functions, constraints, bounds, and solutions of that linear programming formulation. This is demonstrated particularly by the arrows in Table 11, which shows that the network formulation is driven by the constraints at the user nodes, i.e., the lower and upper bounds for the user nodes. These values first influence the equality constraints and subsequently the inequality constraints (these are defined earlier in the paper). The set of solutions in the inequality constraints is directly influenced by the lower and upper bounds for the inequality constraints. Ultimately, the solver ensures that the objective function produces the possible minimum value for the network solution (see the values in the bracket in the first seven columns in the first row of Table 11).

By mapping the coefficients of the linear programming formulation and the network solution in a manner of the Operational Matrix Framework, coefficients that drive the network solution could be identified before incorporating more extensive cost/survivability coefficients as part of meeting an objective function and responding to further constraints. This gave the ship designer awareness, clarity, and confidence in understanding the logical reasoning behind why the solver produces such a network solution. Without the Operational Matrix Framework, the linear programming in the UCL Network Block Approach would not have been sufficiently simplified for early-stage ship systems sizing applications. An overcomplicated network formulation could distract the designer from the main focus of requirement elucidation: to understand the impact of distributed ship service systems (DS3) choices on the overall submarine design, which could have consequences for the vessel's overall architecture. Thus, the Operational Matrix Framework can be seen to reduce the "black box" nature when using the linear programming tool to explore DS3 choices in early stage of ship design.

One of the main areas of future work is to consider whether the existing execution time of the MATLAB script could be further improved by incorporating a new solver other than CPLEX toolbox (e.g., MATLAB *linprog*) to ensure a designer could perform the many iterations required to formulate SUB/RFLOW. Another area would be to expand the application of the Operational Framework to other complex vessels, including but not limited to various surface warships or oil and gas service vessels (Floating Production Storage and Offloading (FPSO) ship, OSVs, drilling ship, etc). This includes expanding the consideration of aspects of maintenance, and supportability for evaluating various submarine systems style choices. The Operational Matrix Framework could also be developed further for investigating the analysis of energy balances for new systems to achieve net zero energy demands for future naval vessels.

CONTRIBUTION STATEMENT

Author 1: conceptualisation, data curation, formal analysis, methodology, writing – original draft. **Author 2:** funding acquisition, supervision. **Author 3:** funding acquisition, writing – review and editing.

ACKNOWLEDGEMENT

This research stems out of a multinational university collaboration involving: University of Michigan, Delft University of Technology, University College London, and Virginia Tech. These partners gratefully acknowledge the funding of this Naval International Cooperative Opportunities in Science and Technology Program (NICOP) contract number N00014-15-1-2752, sponsored by Ms. Kelly Cooper of the US Navy Office of Naval Research, which is a follow-on of a previously successful collaboration. That was undertaken by the first three universities exploring ship layout in early stage of ship design (Andrews et al., 2012). The original work was focused on studying methods for generating and analysing general arrangements in early-stage ship design (Andrews et al., 2012). The second NICOP extended that layout work to study the relationship between distributed system design and general arrangements, in the context of ship survivability (Brefort et al., 2018). The current study could be seen to build on both NICOP projects activities in early stage of ship design research and the current outcome acknowledges the insights from both collaborative endeavours.

DISCLAIMER

The work presented in this paper is part of the thesis of the first author (Mukti, 2022), which in part has been presented in several other papers also referenced in this paper. However, the Operational Matrix focus of this paper has not been presented directly beyond the thesis.

REFERENCES

Andrews, D.J. (2018) The Sophistication of Early Stage Design for Complex Vessels. RINA IJME Special Edition, vol 160 Part A. doi:10.3940/rina.ijme.2018.SE. Oct 2018.

Andrews, D.J., Duchateau, E., Gillespe, J.W., et al. (2012) IMDC State of the Art Report: Design for Layout. 11th International Marine Design Conference (IMDC), Glasgow, UK, 11-14 June, 2012. Available at: http://resolver.tudelft.nl/uuid:f53cb8ba-1174-4975-b12a-c89242027c82 (Accessed: 12 July 2018).

Andrews, D.J. and Pawling, R.J. (2003) "SURFCON A 21st Century Ship Design Tool." In The 8th International Marine Design Conference. Athens, Greece, May 2003. IMDC.

Brefort, D., Shields, C., Habben Jansen, A., et al. (2018) An architectural framework for distributed naval ship systems. Ocean Engineering, Vol 147: 375–385. doi:10.1016/j.oceaneng.2017.10.028.

Brown, A.J. (2020) Design of marine engineering systems in ship concept design. Parsons, M.G. (ed.). Alexandria, VA: The Society of Naval Architects and Marine Engineers. Alexandra, VA. 2020.

Burcher, R. and Rydill, L. (1994) Concepts in Submarine Design. Cambridge ocean technology series. Cambridge [England]: Cambridge University Press. 1994.

IBM, C. (2014) IBM Knowledge Center. Available at: https://www.ibm.com/support/knowledgecenter/SSSA5P_12.5.0/ilog.odms.cplex.help/CPLEX/MATLAB/topics/cplex_matlab_overview.html (Accessed: 29 June 2019).

Kennedy, A.B.W. and Sankey, H.R. (1898) The Thermal Efficiency of Steam Engines. Report of The Committee Appointed to The Council Upon the Subject of The Definition of a Standard or Standards of Thermal Efficiency for Steam Engines: With an Introductory Note. (Including Appendixes and Plate at Back of Volume). Minutes of the Proceedings of the Institution of Civil Engineers, 134 (1898): 278–312. doi:10.1680/imotp.1898.19100.

MATLAB (2019) MATLAB - MathWorks. Available at: https://uk.mathworks.com/products/matlab.html (Accessed: 8 July 2019).

Mukti, M., Pawling, R. and Andrews, D. (2024) Computer Aided Sketching in The Early-Stage Design of Complex Vessels. Ocean Engineering. To be Published

Mukti, M.H. (2022) A Network-Based Design Synthesis of Distributed Ship Services Systems for a Non Nuclear Powered Submarine in Early Stage Design. Ph.D. thesis, UCL (University College London). Available at: https://discovery.ucl.ac.uk/id/eprint/10147800/ (Accessed: 21 November 2022).

Mukti, M.H., Pawling, R.J. and Andrews, D.J. (2021) Distributed Ship Service Systems Architecture in The Early Stages of Designing Physically Large and Complex Vessels: The Submarine Case. IJME, Vol 163. doi:https://doi.org/10.5750/ijme.v163iA2.755. 2021.

Mukti, M.H., Pawling, R.J. and Andrews, D.J. (2022) "Development of an Early-Stage Design Tool for Rapid of Distributed Ship Service Systems Modelling in Paramarine – a Submarine Case Study." In 21th Conference on Computer Applications and Information Technology in the Maritime Industries. Pontignano, Italy, June 2022.

Mukti, M.H., Pawling, R.J. and Andrews, D.J. (2024) Computer Aided Sketching in The Early-Stage Design of Complex Vessels. Ocean Engineering Journal. To be Published.

Newman, M. (2010) Networks: An Introduction. Oxford University Press. Available at: http://www.oxfordscholarship.com/view/10.1093/acprof:oso/9780199206650.001.0001/acprof-9780199206650 (Downloaded: 22 November 2018).

Parsons, M.A., Kara, M.Y., Robinson, K.M., et al. (2020) Early-Stage Naval Ship Distributed System Design Using Architecture Flow Optimization. Journal of Ship Production and Design, 37: 1–19. doi:10.5957/JSPD.10190058. 2020.

Qinetiq (2019) Paramarine. Available at: https://paramarine.qinetiq.com/products/paramarine/index.aspx (Accessed: 8 July 2019).

Robinson, K.M. (2018) Modeling Distributed Naval Ship Systems Using Architecture Flow Optimization. Master's thesis, Virginia Tech. 2018.

Trapp, A. (2015) Shipboard Integrated Engineering Plant Survivable Network Optimization. Ph.D. thesis, Massachusetts Institute of Technology. 2015.

APPENDIX A

```
G = digraph([0 1 1 0; 0 0 0 1; 0 0 0 1; 0 0 0 0]);
G.Nodes.Name = {'Source' 'Hub1' 'Hub2' 'Target'}';

ops_mtx=[1 1 2 2 0 0 0 0;
        1 0 1 0 -1 0 0 0;
        -1 1 0 0 0 -1 0 0;
        0 0 -1 1 0 0 -1 0;
        0 -1 0 -1 0 0 0 -1;
        0 0 0 0 0 0 0 -10;
        10 10 10 10 inf 0 0 -10];
x=linprog(ops_mtx(1,:),[],[],ops_mtx(2:5,:),zeros(4,1),ops_mtx(6,:),ops_mtx(7,:));

G.Edges.Weight(1) = x(1,1);
G.Edges.Weight(2) = x(3,1);
G.Edges.Weight(3) = x(2,1);
G.Edges.Weight(4) = x(4,1);

G.Edges.LWidths = 7*G.Edges.Weight/max(G.Edges.Weight)+1; p=plot(G); p.LineWidth =
G.Edges.LWidths;
```