Algorithmic Extraction of Power Flow Configurations for Three-Port DC-DC Converters

Georgios Salagiannis
Department of Electrical and Computer Engineering
University of Patras
Patras, Greece
gsalagiannis@ac.upatras.gr

Emmanuel Tatakis
Department of Electrical and Computer Engineering
University of Patras
Patras, Greece
e.c.tatakis@ece.upatras.gr

***Abstract*—During the last two decades, power flow graphs have been established as a systematic and straightforward method for the design of DC-DC power converters. Several researchers have focused their efforts on the design of Three-Port Converters with this method for the integration of renewable energy sources, energy storage systems and DC micro-grids. However, even though all attempts utilize the same design method with slightly different application specifications (ESS existence, output bidirectionality, etc.), there is currently no attempt to group all the available power flow configurations under one procedure. This gap is addressed in this paper, where a systematic and algorithmic method is proposed to automatically derive all DC-DC converter power flow configurations, based on rule application and a single-variable user input. This work not only highlights the fundamentals of the power flow graph design while automating it, but also sets the ground for further study and optimization of a plethora of different configurations, without the restrictions posed by traditional manual design.**

***Index Terms*—DC-DC Converters, Graph-Theory, Renewable Energy, Power Flow Graphs, Three-Port Converters**

I. Introduction

The Power Flow Graph (PFG) converter design was introduced by [1]. Initially, it was used to systematically derive converter topologies for Power Factor Correction (PFC), where an Energy Buffer (EB) was used. This led to the invention of Reduced Redundant Power Processing (R^2P^2) converters, which were Three Port Converters (TPCs), interconnected through basic converters. The main idea was that by changing the wiring configuration of those converters, higher system efficiency compared to a cascaded connection could be achieved.

This design approach can be implemented in a plethora of applications, where the analysis based on power flow graphs applies. Yang et al. [2] utilized the R^2P^2 concept and power flow graphs to systematically examine and derive new topologies for Double Input Single Output (DISO) DC-DC converters. The researchers distinguish two cases; a) both inputs are voltage sources and b) one input is a voltage source and the other one is an Energy Storage System (ESS). For both cases, only the graph configurations allowing simple interconnections (meaning that at least one Type I sub-graph is existent) are discussed. For the circuit implementation, unidirectional (bidirectional) basic converters (buck, boost and buck-boost) are used for the first (second) case. The primary evaluation criterion is the efficiency of the overall system.

Provided the increasing interest for DC-DC converters with high step-up capability, Palomo et al. [3] attempted to design converters with quadratic voltage boosting with non-cascaded configurations, in order to address the inherently low efficiency of the cascaded quadratic boost family of converters. In this work, the TPC consists of an input, an EB and an output port, and only unidirectional basic converters are considered. Two fundamental contributions are made to the field; a) the developed configurations are systematically compared based on the graph configuration inherent gain and b) the concept of topology simplification and switch number minimization in R^2P^2 is introduced. Notably, only Types I-II and I-III are examined, owing to their wide conversion ratio.

Regardless of the previous efforts, a study and record of all possible R^2P^2 configurations and their suitability for high step-up applications was nonexistent. Therefore, this gap was filled by [4], for converters with an input, an EB and an output port. All potential configurations were examined for converters composed of non isolated unidirectional basic converters and both their efficiency and voltage gain were derived.

During last decade, distributed power generation and DC Microgrids have emerged as a promising solution towards energy sustainability. In this direction, hybrid integration of RES and ESS is required, which has led to the development of TPCs with controlled storage port (opposite to the previous attempts with EB) [5]. Instead of a Single Input Single Output (SISO) converter for each component, a unified converter is used. In [6], based on the idea that TPCs can be composed from basic converters, power flow graphs are used to create and analyze TPCs for DC Microgrid applications. Specifically, two bidirectional ports are considered, namely the ESS and the output DC Bus. Opposite to [4], the storage port is not considered as an EB freely satisfying power conservation, but rather as a fully controlled ESS. This leads to the utilization of both unidirectional and bidirectional converters. However, no efficiency or gain analysis is performed.

The current status of the research on the field of PFG Three-Port DC-DC converter design is summarized in Table I. It becomes apparent that even though some systematic derivation

The present work was financially supported by the Andreas Mentzelopoulos Foundation (Corresponding Author: G.S.).

TABLE I: Literature Review on PFG Three-Port DC-DC Converter Design.

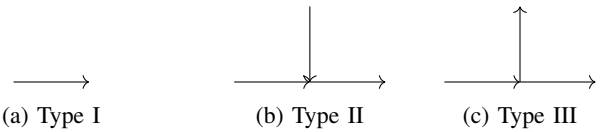
Researchers	Year	#Ports	# ↔ Ports	Contribution
Tse et al.	2001	3	1	Systematic study of power supplies with integrated PFC capability as TPCs, using power flow graphs and unidirectional fundamental converters.
Yang et al.	2015	3	1	Systematic study of TPCs with no or one bidirectional port (ESS), using power flow graphs and unidirectional/bidirectional fundamental converters.
Palomo et al.	2015	3	1	Derivation of non-cascaded high step-up DC-DC converters, using power flow graphs and unidirectional fundamental converters.
Zogogianni et al.	2019	3	1	Systematic study of TPCs with one bidirectional port (Energy Buffer), using power flow graphs and unidirectional fundamental converters.
Aljarajreh et al.	2021	3	2	Systematic study of TPCs with two bidirectional ports (ESS and Output), using power flow graphs and unidirectional/bidirectional fundamental converters.
Proposed	2024	3	1-2	Algorithmic derivation of power flow configurations for TPCs with EB and TPCs with ESS and unidirectional or bidirectional output, using both unidirectional and bidirectional converters.

methods are described, there is no universal method to cover all possibilities (TPC with EB, TPC with ESS bidirectional port and TPC with bidirectional ports for ESS and output). Furthermore, manual effort is required to derive and examine all available power configurations, and therefore some studies are only focused on the graphs incorporating simple interconnections. In this paper, those issues are addressed by developing a programmable method for the derivation of Three-Port DC-DC converters with different features, based on the input provided by the user. Based on that, the contributions of this paper are summarized as follows:

- It presents a coherent way to translate the traditional drawn power flow graphs to matrices that can be processed by a PC, so that the digital forms of PFGs with FCs can be generated.
- It proposes a universal and programmable method for the derivation of power flow configurations of TPCs that permits the design of any relevant converter only by changing certain parameters.
- It demonstrates and implements the algorithmic rationale for TPC converter design, summarizing previous works and providing useful insight to the researchers.

II. Background for PFG Converter Design

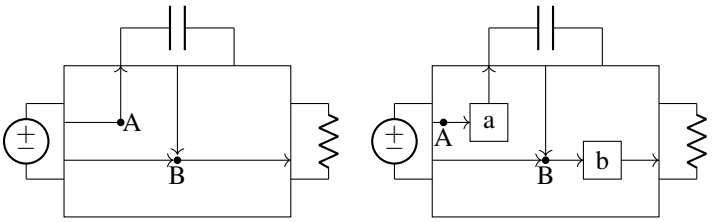
As described above, converters' PFG design is based on the idea that simple converters can be interconnected in a way that only part of the total power is processed through them. To concisely demonstrate various interconnection options, the power flow graphs were proposed. Based on the nature of its source or load, a port can either exclusively receive or offer power if it is unidirectional, or both if it is bidirectional. Furthermore, one port may offer (receive) power to (from) multiple ports. As shown in Fig. 1, **Type I** is when power is transferred from one port to another, **Type II** is when two ports provide power to a third port and **Type III** is when a port provides power to the other two ports. These SubGraphs (SGs) are combined to indicate all the power flow paths in a TPC, forming the complete PFG. Each arrow indicating power flow from (to) one port to (from) the SG will be called power flow vector from now on. For the case of a converter with input, EB and output ports, the complete power flow graph of Type I-IIA is shown in Fig. 2a as an example. In the next design step,



(a) Type I (b) Type II (c) Type III

Fig. 1: The Three Available SG Types.

simple converters that will be called Fundamental Converters (FCs) from now on and will be denoted by lowercase letters a, b, etc., are interfered on the PFGs to achieve power flow controllability (Fig. 2b).



(a) PFG (b) PFG with incorporated FCs

Fig. 2: Power Flow Graphs for Type I-IIA configuration.

III. TPC Derivation with Programmable Method

Most configurations have already been presented in the literature. However, no attempt thus far has taken advantage of the similarity of the derivation method, to unify the design procedure and therefore this is attempted here. The method consists of three main parts as shown in Fig. 3; the PFG to Matrix conversion, the creation of all possible PFGs and the FC placement on them to create the power flow configurations.

A. PFG to Matrix Conversion

By using graph theory principles, the problem can be easily understandable by a computer [7]. The first step towards this direction is to convert the PFGs, similar to the ones in Fig. 2a, into proper graphs. To accomplish this, a graph G(V, E) is created, where the vertex set V contains all the power ports (1→RES, 2→EB/ESS, 3→Output) of the TPC and the SGs (A, B, etc.) that form it, while the edge set E contains all the power flow connections between the power ports and the SGs of the system. Because the power flow is a bidirectional parameter,

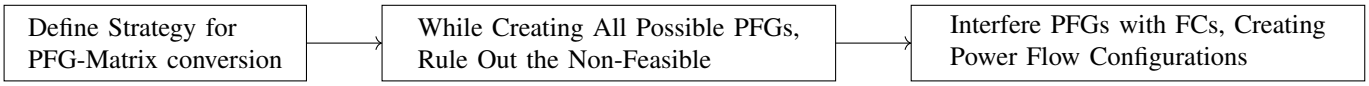


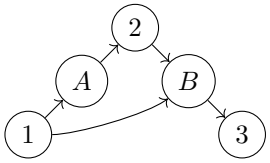
Fig. 3: Algorithmic Process for the derivation of TPC power flow configurations.

the graph must always be a digraph. Based on those, one can deduct the digraphs of the power flow graphs, as shown in Fig. 4a for the case of the PFG of Fig.2a.

However, a digraph is still a graphical representation of the system. To be able to transform it in a data structure, usually an adjacency matrix is used [8]. Instead of using the traditional adjacency matrix, a specialized and compact adjacency matrix is introduced, namely the Power Flow Graph Adjacency Matrix (PFGAM). The rows of the PFGAM are the input and output ports of the TPC, while the columns indicate the SGs used. Therefore, the size of the two-dimensional matrix is $\mathbf{3} * \mathbf{N_{SG}}$, where N_{SG} is the number of sub-graphs composing the power flow graph. Each cell of the PFGAM may contain one of the following values:

$$x_{i,j} = \begin{cases} 1, & \text{if port i is an input to SG j} \\ -1, & \text{if port i is an output to SG j} \\ 0, & \text{if port i is not connected to SG j} \end{cases} \quad (1)$$

Therefore, the digraph in Fig.4a can be translated in the PFGAM of Fig. 4b and every possible configuration can be represented as a PFGAM in a unique manner.



(a) Graph

	A	B
1	1	1
2	-1	1
3	0	-1

(b) PFGAM

Fig. 4: Computer-friendly version of Type I-IIA PFG.

B. Feasible PFGAM Set Creation

The fundamental feature of this algorithm is that no previous knowledge on feasible PFGAMs is required by the user or the program itself. The user only needs to select the value of the variable **Bidi_Ports**; 0 for TPC with EB, 1 for TPC with ESS and 2 for TPC with ESS and bidirectional output. Therefore, during the first step of the execution, each cell of the PFGAM takes all the possible values (column by column) to create all the possible combinations. Obviously, most of them cannot be implemented in the real world. Therefore, a set of rules is applied on the PFGAM to clear all the non-feasible matrices.

1) Rule 1 (Port Bidirectionality):

A) The input (RES) port only produces power and therefore the power flow is unidirectional. Consequently, the elements of the first row (RES port) may have a value of 0 or 1. Additionally, the ESS or EB port both offers and receives power based on the current operation of TPC. Consequently, each cell in the second row can take any of the three values (-1, 0, 1). The output port can either be unidirectional where its cell value set is {-1, 0} or bidirectional where the value set is full.

B) Each bidirectional port must have at least one intake and one outtake of power. Equivalently:

- The sum of the absolute values of all row elements of any bidirectional row must be greater than the absolute value of the sum of all row elements.

Proof: Sum(ABS) cannot be lesser than ABS(Sum). Supposing that Sum(ABS) is equal to ABS(Sum), the cell values of the port row have the same sign. Therefore, all connected SGs are either providing or consuming power from the port, which is not acceptable.

2) Rule 2 (SG I/O Existence): Similar to Rule 1B, each one of the nodes that represent an SG, must have at least one input and one output, therefore:

- The sum of the absolute values (Sum(ABS)) of all column elements of each SG column must be greater than the absolute value of the sum (ABS(Sum)) of the same column elements.

3) Rule 3 (SG Interleaving Prohibition): Some of the following rules, need to have information on the exact position (row, column) of each connection to its SG. To obtain a compressed version of this information, an n-digit numerical system can be used, where n is the number of all potential PFGAM cell values (3 in our case). Therefore, in this paper a ternary numerical system is used. Then, the declaration of the positions of LSB and MSB is performed. For column-wise encoding, the data is compressed column-wise and the LSB is the cell element of the first row. Therefore, a $3 * SG$ PFGAM column-wise compression will produce an $1 * SG$ matrix.

Specifically for Rule 3, interleaving of SGs is not permitted as this not constitutes a different power flow configuration. The researchers should decide later on the interleaving of the SGs based on the final converter topology and on power level.

- The columns of a PFGAM should have a unique (different among each other) compressed column.

4) Rule 4 (Use of Minimum Interconnections): The aim of both this study and the previous ones is to implement every power flow configuration with the minimum number of SGs and FCs, hence there has to be no overlap in the power flows.

A) To guarantee that, the algorithm follows the rule below, which was formulated by observing the resulting configurations of the previous works [2],[4],[6].
The number of non-Type I SGs must be lower or equal to the maximum potentially required non-Type I SGs. This limit is based on the number of total SGs (h) and the number of Bidirectional Ports (Bidi_Ports) and it is expressed as:

$$max\_non\_I[h] \quad (2)$$

where $max\_non\_I[h]$ is a list with the range limits:

$$max\_non\_I = [Min.SGs - Bidi\_Ports, Min.SGs] \quad (3)$$

and $Min.SGs$ is the minimum required number of SGs which is equal to two for all cases.

B) If two SGs have identical power flow paths, then this will be detected by Rule 3 and the PFGAM will be cleared out. However, a Type I SG in parallel with one branch of a Type II or III SG will not be filtered and therefore an additional rule is needed.
   - There shouldn't be a pair of any SGs, where any two cells in the same row are equal and non zero, provided that exactly one SG is Type I.

5) Rule 5 (PFGAM Uniqueness): PFGAMs with the same configuration and interchanged names among the SGs are not considered unique. Therefore:

- The compressed columns of a PFGAM should not be identical with any other PFGAM, even if the columns are interchanged in any way.

The first part of the algorithm is developed so that all feasible PFGAMs will be extracted. The algorithm is presented in a pseudocode form in Fig.5 and explained below:

1) Generate all possible PFGAMs: For a given set of cell values {-1,0,1}, all possible PFGAMs can be generated by starting from a $\mathbf{3 * N_{SG}}$ zero matrix X and systematically changing the value of each $x_{i,j}$ to produce all the combinations. The number of those combinations is $3^{3*N_{SG}}$, which means that it becomes huge even for small values of N_{SG} ($N_{SG} = 3 \longrightarrow 19683$ combinations). This number can be significantly reduced by utilizing Rule 1A. Specifically, when creating the combinations for the first row (input port), the cell value -1 should not be included in the set of cell values, as the input port only produces power. Similar goes for the rest of the ports. By implementing those constraints the available combinations are reduced to $2^{2*N_{SG}} * 3^{N_{SG}}$ ($N_{SG} = 3 \longrightarrow 192$ for the case of a single bidirectional port (ESS)).
2) Algorithm Optimization: The possible PFGAMs are generated column-by-column and the results are included in the set of PFGAMs so that they will be used during the generation of next column. When this happens, the initial results are obsolete (as they are included in the next results with the information of one more column included). Therefore they can be removed, prior to the column's rule check to reduce execution time.
3) Rule Application to the PFGAMs: Most of the suggested rules are applied, by definition, column-wise on the PFGAM. Therefore, the approach of PFGAM creation column-by-column has significant advantage as Rules 2, 3, 4, 5 are applied after the creation of each column, clearing out invalid PFGAMs prior to their full creation. On the contrary, Rule 1B is applied row-wise and consequently the complete PFGAM must be available.

```
Initialize empty PFGAM set;
Initialize 3 × SG PFGAM with zeros;
Insert it in the Set;
for each column (j) of PFGAM do
    for each row (i) of PFGAM do
        for each available PFGAM(l) in PFGAM Set
         do
            for each available cell value (k) do
                PFGAM_Set[l, i, j] ←
                 cell_values[k];
    /*Column j has been created*/;
    Remove initial PFGAMs from Set;
    Apply Rules 2, 3, 4, 5;
/*All cells have been updated*/;
Apply Rule 1B;
Find the type of each SG;
Find position of bidirectional SGs;
```

Fig. 5: Pseudocode for Feasible PFGAM Set Creation

Following this procedure, all feasible PFGAMs are created and the next step of the method can be executed.

C. FC Placement in PFGs

For this step, the structure of the initial PFGAMs created in the previous section will be updated as shown in result Figs. 9, 10 and 11. More specifically, three columns (one for each port) will be added right of the initial PFGAM. Additionally, below the initial PFGAM as many rows as the number of FCs needed will be added. This is necessary in order to correctly display the connections of different elements after the FC placement. Cells indicating connections between SGs and ports get zeroed now, if an FC is placed between the port and the SG. If a port both receives and offers power through the same bidirectional converter, a high value (10) is inserted in the corresponding cell, as there isn't any simple way to insert ± 1 in an array. This part of the algorithm is presented in a pseudocode form in Fig. 6 and is described below.

1) Determination of FC positions: First of all, the SGs are separated in two different data pools (uni-pool and bidi-pool), based on their suitability for unidirectional or bidirectional FC placement. By definition, one SG cannot be bidirectional. Therefore, for a bidirectional FC to be utilized, two fitting SGs must be combined. By SG fitness it is meant that two SGs must have exactly two power flow vectors that are complementary, i.e. they exhibit opposite power flow. To search up for these SGs, the procedure below can be followed:

a) For the case of TPC with EB, no bidirectional FC will be used and therefore this procedure is bypassed.
b) All possible combinations of SG pairs are created.
c) For the rest of the cases, a 2*2 array is created for each SG pairs. The first row of the array elements consist of

```
uni_pool ← unidirectionalSGs;
bidi_pool ← bididirectionalSGs;
/*ports where a Bidi FC can be placed*/;
pool ← [2, Bidi_Ports + 1];
inv_pool ← [1, 3] − [1, 3] ∩ pool;
if no Type I SG exists then
    for each port (i) of pool do
        for each port (j) of inv_pool do
            set ← [set, i, j] /*assign ports(i, j) as FC
              placement pair*/;
        for each port (j) of pool do
            if 1 then
                set ← [set, i, j] /*assign ports(i, j) as
                  FC placement pair*/;
else
    set ← 2;
    for each non Type I bidirectional SG (j) do
        for each port (i) do
            if port i receives power then
                set ← i
for each combination i of set do
    configuration ← zeros(column_number(x) −
    length(bidi_pool), column_number(x) + 3);
    for each element j of current combination i do
        for each bidirectional SG k do
            y(j, bidi_pool[k]) ←
              x(set[i, j], bidi_pool[k]);
            y(j, SG + set[i, j]) ←
              −x(set[i, j], bidi_pool[k]);
    Store y in the power flow configurations set;
uni_port ← Create_uni_port();
for each unidirectional SG (j) do
    for each port (i) in uni_port connected to SG j do
        for each PFG (k) with updated previous
          columns do
            y(length(bidi_pool)/2+j, uni_pool(j)) ←
              x(i, uni_pool(j));
            y(length(bidi_ind)/2+j, size(x, 2)+i) ←
              −x(i, uni_pool(j));
```

Fig. 6: Pseudocode for Power Flow Configurations Creation

the corresponding elements (same SGs) of the ESS row of the PFGAM, while the second row consists of:

- Bidirectional Output: the corresponding elements of the output row of the PFGAM.
- Unidirectional Output: the first row multiplied by -1.

Cascaded configuration Type I-I is excluded from the process, as in this case, even though there is a bidirectional port, no bidirectional FC is used.

d) The sums of the primary and secondary diagonal of the array are calculated.

e) If those sums are opposite to each other and the absolute sum of each is greater than one, then this SG pair may be interfaced with a bidirectional FC. This is because that condition guarantees that both rows offer and receive power complementary to both SGs and themselves, which means that a bidirectional power path is existent.

Furthermore, the range of ports where a bidirectional FC can be placed is determined from the range below:

$$Bidi\_FC\_Ports\_pool = [2, Bidi\_Ports + 1] \tag{4}$$

This equation summarizes that in case of TPC with EB, no bidirectional FC is placed, while in case of a TPC with ESS either port 2 (Unidirectional output) or ports 2 and 3 (Bidirectional output) can be interfaced with a bidirectional FC. The complementary set (inv_Bidi_FC_Ports_pool) is also created, as it is used for the creation of ports pairs where FCs will be placed.

2) Bidirectional FC Placement: Diving into the potential placement of bidirectional FCs, the algorithm checks if a Type I SG is existent in the PFG. **If not**, no SISO converter will be used, as each FC will either be bidirectional or will be used for the joint connection of a port to two SGs as shown in Fig. 7 of [6]. In this case, it is crucial to determine in which ports will the FCs be connected to. Depending on the number of required FCs, connection pairs are created both between the Bidi_FC_Ports_pool and the inv_Bidi_FC_Ports_pool elements, but also between the Bidi_FC_Ports_pool elements themselves. Then, for each created combination set, a bidirectional FC is placed into each port of the set. The FCs are placed in between the bidirectional SGs (bidi_pool). **Alternatively**, SISO converters will be used and the only bidirectional FC will be connected between SGs contained in bidi_pool and the first port (by convention) of Bidi_FC_Ports_pool.

3) Unidirectional FC placement: At this point, the placement of the bidirectional FCs is finished. Now, one unidirectional FC needs to be placed in each unidirectional SG to achieve full controllability of the system. The Create_uni_port function is called, having the following functionality; if uni_pool is empty but a Type I SG is existent, that means that there are only two SGs. Apparently, a unidirectional FC should not be placed on the port that has the bidirectional power flow, so a list called uni_port is created with the rest of the ports where a unidirectional FC is acceptable. The FC can be placed between any port of uni_port that is connected to this SG and the point X{A, B, etc.} of intersection of power flow vectors. If the SG is Type-I, there is no need to put FCs in all ports, as there are only two ports involved and placing the FC in one port is identical to placing the FC in the other port. Similar to the initial PFGAM creation, the extended PFGAM is updated column-by-column (SG-by-SG).

IV. Case Study

To further clarify the procedure described above, a case study will be performed in this section, where a PFG will be extracted. For the sake of completeness, a PFG with two bidirectional ports ($Bidi\_Ports = 2$) will be examined so that every part of the algorithm is described. The PFG examined is **Type I-III-I**, which will lead to the power flow configurations

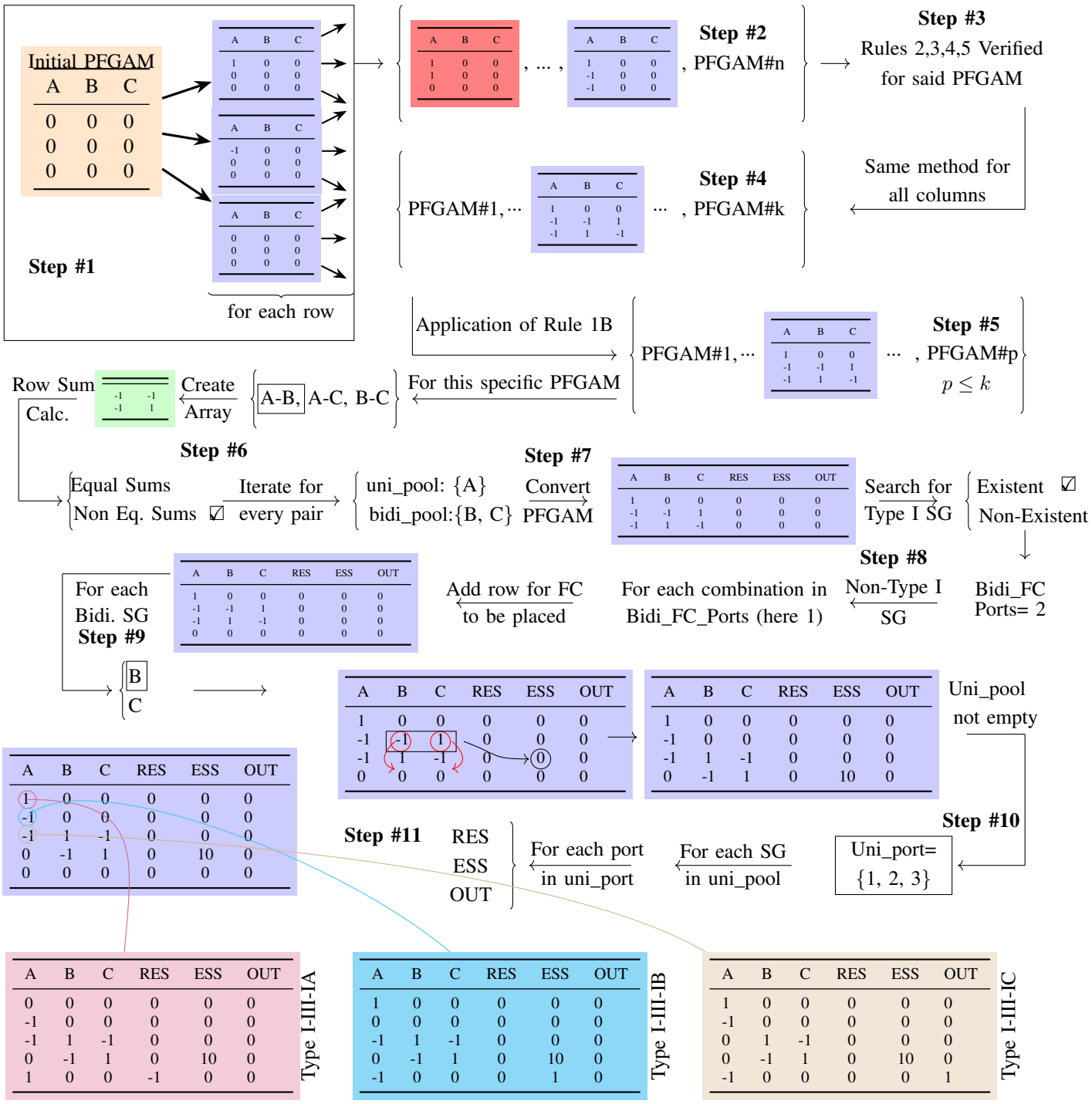


Fig. 7: Abstract display of the algorithm's execution, focused on Type I-III-I PFG.

I-III-IA, I-III-IB and I-III-IC. The algorithm execution for this case study is depicted in Fig. 7, where different steps are denoted with **Step #** and arrows indicate the serial execution of the algorithm.

This PFG will be derived from the first part of the algorithm. Initially, based on the $Bidi\_Ports$ parameter, the number of the required SGs is computed. For $Bidi\_Ports = 2$, SGs may be between 2 and 4. Starting with 2, all feasible PFGs are derived. Similarly, for $SG = 3$, a $3 * 3$ matrix is created and filled with zeros. Now, for the first column, one value of the set {1,-1,0} is inserted in the cell of the first row. The altered PFGAM (with only one cell different) is stored in the set of potential PFGAMs, without deleting the initial $3 * 3$ zero PFGAM used for its creation. This is repeated for the remaining values of the set (**Step #1**). The procedure is then performed for the cell of the second row (while still

being in column 1), with the difference that now there isn't only one PFGAM where the cell value must be changed, but the operation must be done for all PFGAM instances (zero PFGAM, plus 3 cases created for the previous cell). It becomes apparent that the possible PFGAMs increase in number in each iteration. By carrying on this procedure, all combinations of first column values are stored in different PFGAMs (**Step #2**). One of these cases is the one that we are interested to, which is displayed in Fig. 7 in its current form. As it was stated in the previous section, a PFGAM set number reduction is performed after each column, by removing the initial PFGAMs that were used to update the set. For the first row, only the initial full zero PFGAM must be deleted. For graph simplicity, this procedure is not depicted in the relevant figure.

Prior to moving on the second column (SG), the rule application for the rules that are applied column-wise must be performed, to clear every non-feasible PFG (**Step #3**). By simple observation it can be seen that the Type I-III-I PFGAM complies with Rules 2, 3, 4, 5. On the contrary, for reference purposes, the PFGAM displayed in Fig. must be deleted as it violates Rule 2. After rule application, the same procedure is followed for the rest of the columns (**Step #4**). Then, the complete PFGAMs are formed and the last Rule 1B must be checked (row-wise). Again, the PFGAM of interest is compliant (**Step #5**).

Moving to the second part of the algorithm, the suitable FCs must be placed in each PFGs. The first step is to separate the SGs suitable for unidirectional and bidirectional FC placement in two different pools (uni_pool and $bidi\_pool$, **Step #6**). The possible combinations of SG pairs are created (A-B, A-C, B-C). For the first pair (A-B) the array created is shown. The sum of the primary diagonal is **0**, while the one of the secondary is **-2**. The two sums are not equal, meaning that the two SGs cannot form a pair for a bidirectional FC placement. The same method is followed for the rest of the SG pairs leading to $uni\_pool = \{A\}$ and $bidi\_pool = \{B, C\}$. According to Eq. 4, in our case, the bidirectional FC can be placed in the power path of either the ESS or the output port.

Having separated the bidirectional and unidirectional SGs, the FC placement starts with the bidirectional one. To store the additional information of how the FCs are connected to the PFGs, one additional column for each port and one extra row for each FC is added to the PFGAM (**Step #7**). The row will be added one at a time, as the algorithm decides during the execution the number of FCs required for each PFG. To determine in which ports will the FC be placed, the following procedure is followed (**Step #8**). Checking whether the PFGAM contains any Type I SG, two are found. Therefore, one bidirectional FC will be placed between the two bidirectional SGs. For contractual reasons, it is selected that the FC is placed on the port that is first encountered by the algorithm (here ESS port). Prior FC placement, a row is added to the PFGAM. For each bidirectional SG, the FC is inserted between the SG and the corresponding port, leading to the zeroing of the relevant cells and the transfer of their original values in the row of the FC (**Step #9**), as shown in Fig. 7.

Then, one unidirectional FC will be placed in each unidirectional SG. The $Create\_uni\_port()$ function is called, but the PFGAM of interest does not belong to the special cases addressed by the function, so $uni\_ports = 1, 2, 3$ (**Step #10**). For the first and only unidirectional SG (A), an FC is placed in one port of the uni_port each time, creating three different cases; Type I-III-IA, I-III-IB and I-III-IC (**Step #11**).

V. Results

Based on the developed algorithm, all possible PFG configurations can be extracted for three different system options, namely the TPC with EB, and TPC with ESS and unidirectional/bidirectional output. The configurations are discussed in the following subsections, where indicative results are shown. To further increase ease of use of the algorithm, a graphical user interface was developed in MATLAB. Via this app, the user can choose between finding the PFGs with or without the corresponding FCs, for the required system characteristics. The results are displayed one at a time both in their PFGAM and their graph forms. In Fig. 8, the app is shown presenting the resulting PFG of type I-III-IA, which was examined in the case study.

All results can be found in the relevant **Github repository**. The **execution time** has an order of magnitude of **1s**.

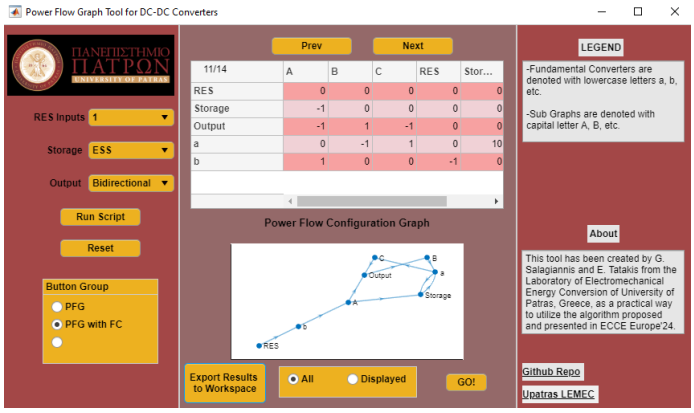


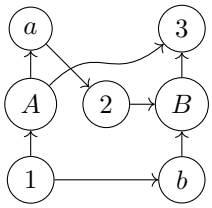
Fig. 8: The developed tool in MATLAB.

For the design of TPCs with EB, only unidirectional FCs are required. Furthermore, no control is required for the power flow of the EB, leading to a direct (without the interference of a converter) connection between that and parts of the rest of the system. The full set of available configurations have been initially grouped and studied by [4]. In this paper, all those PFGs were generated by the proposed algorithm, with one configuration presented in Fig. 9. In this Figure, both the PFGAM and the corresponding graph are shown. The gray coloured cells do not have any essential meaning, as two ports cannot exchange power directly. The red coloured cells indicate the connections on the initial PFGAM that need to be deleted (zero inserted) after the FC placement.

For ESS integration, a bidirectional FC needs to be placed between the ESS port and the rest of the system, with the

	A	B	1	2	3
1	1	1			
2	-1	1			
3	-1	-1			
a	-1	0	0	1	0
b	0	1	-1	0	0

(a) Type IIA-III PFGAM

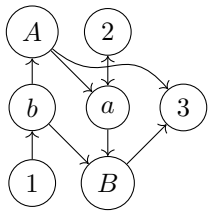


(b) Type IIA-III Graph

Fig. 9: Possible configuration of TPC with EB

	A	B	1	2	3
1	1	1			
2	-1	1			
3	-1	-1			
a	-1	1	0	10	0
b	1	1	-1	0	0

(a) Type II-IIIA PFGAM



(b) Type II-IIIA Graph

Fig. 10: Possible configuration of TPC with ESS

exception of the Type I-I PFG. Some available configurations have been initially grouped and studied by [2]. Notably, only PFGs with at least one Type I SG were presented. On the contrary, in this paper, all feasible PFGs were generated by the proposed algorithm, with some results shown in Fig. 10.

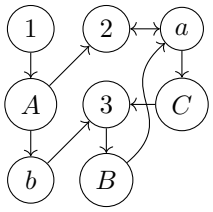
To incorporate both an ESS and a bidirectional output, a bidirectional power path needs to be established between the two ports. This can be achieved either with one or two bidirectional FCs, depending on the PFG configuration. The full set of available configurations have been initially grouped and studied by [6], with the exception of PFGs I-IIA and I-IIB. In this paper, all PFGs were generated by the proposed algorithm, with some results presented in Fig. 11.

VI. Conclusion

In this paper, an algorithm is developed, aiming to automate the PFG design process for Three-Port DC-DC power converters. The methods for translating the problem to a language understandable by a PC, validating the generated PFGAMs and placing the FCs in PFGs are properly described and the corresponding algorithm is presented in the form of pseudocode. Subsequently, a case study and several generated results (PFGAMs and graphs) are presented and discussed. Throughout this study, its contribution on the field is highlighted, as it does not only sets the PFG design methodology in a structured way, but also releases it from the restrictions and the toil associated with the manual derivation process. In future works, additional processing steps should be added after this algorithm to include optimization studies based on specific parameters, i.e. voltage gain and efficiency, and/or discover the most fitting power converter topologies for the FCs that consist the TPC converter.

	A	B	C	1	2	3
1	1	0	0			
2	-1	-1	1			
3	-1	1	-1			
a	0	-1	1	0	10	0
b	-1	0	0	0	0	1

(a) Type I-III-IC PFGAM



(b) Type I-III-IC Graph

Fig. 11: Configuration with ESS and Bidirectional Output

References

[1] C. Tse and M. Chow, "Theoretical study of switching power converters with power factor correction and output regulation," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 7, pp. 1047–1055, Jul. 2000.

[2] P. Yang, C. K. Tse, J. Xu, and G. Zhou, "Synthesis and analysis of double-input single-output DC/DC converters," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 10, pp. 6284–6295, Oct. 2015.

[3] R. Loera-Palomo and J. A. Morales-Saldaña, "Family of quadratic step-up dc–dc converters based on non-cascading structures," *IET Power Electronics*, vol. 8, no. 5, pp. 793–801, 2015.

[4] C. G. Zogogianni, E. C. Tatakis, and M. S. Vekic, "Non-isolated reduced redundant power processing DC/DC converters: A systematic study of topologies with wide voltage ratio for high-power applications," *IEEE Transactions on Power Electronics*, vol. 34, no. 9, pp. 8491–8502, Sep. 2019.

[5] G. Salagiannis and E. Tatakis, "Review on non-isolated multiport converters for residential DC microgrids," *Energies*, vol. 17, no. 1, p. 222, Jan. 2024.

[6] H. Aljarajreh, D. D.-C. Lu, Y. P. Siwakoti, C. K. Tse, and K. W. See, "Synthesis and analysis of three-port DC/DC converters with two bidirectional ports based on power flow graph technique," *Energies*, vol. 14, no. 18, p. 5751, Jan. 2021.

[7] Y. Li, J. Kuprat, Y. Li, and M. Liserre, "Graph-theory-based derivation, modeling, and control of power converter systems," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 10, no. 6, pp. 6557–6571, Dec. 2022.

[8] F. Dörfler, J. W. Simpson-Porco, and F. Bullo, "Electrical networks and algebraic graph theory: Models, properties, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 977–1005, May 2018.