A methodology for distributed fault diagnosis

V. Gupta, V. Puig and J. Blesa

1 Advanced Control System Research Group, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain

Abstract. In this paper, a methodology for distributed fault diagnosis is proposed. The algorithm places the sensors in a system in such a manner that the partition of a system into various subsystems becomes easier facilitating the implementation of a distributed fault diagnosis system. This algorithm also reduces or minimized the number of sensors to be used or install thus reducing overall cost. Binary integer linear programming is used for optimization in this algorithm. Real case study of Barcelona water network has been used to demonstrate and validate the proposed algorithm.

1. Introduction
The model based diagnosis is all about diagnosing faults in the system by comparing the set of system observations with a model. A suitable approach basically depends on the system model and the faults to diagnose since there is no general framework for such diagnosis systems. In any time during their working life the processes can suffer from faults or malfunctions. A diagnosis system works in parallel with the process. Observations provided by the process measurements and the control or operating signals are the diagnosis system inputs. The diagnosis system tries to infer which behaviour described by the model is actual in the process from these observations.

Serious damage on systems or a loss of process performance can be prevented by diagnosing faults correctly. This makes fault diagnosis [1] an important research area but it is rare to find novel techniques in industrial processes. The main reasons are first that the achieved diagnosis performance is not the expected one and the second is, design techniques are cumbersome and difficult to implement. The present work tries to solve or simplify these problems by choosing the location of sensors in the process. Generally performance of diagnosis systems strongly depend on the on-line information acquired from the process. Therefore, appropriate sensor location will lead to better implementation facilities and diagnosis performance.

Sensors are the components of the system by means of which process observations are obtained. Therefore, the number of observations increases and consequently so does the analytical redundancy degree of the model by installing extra sensors in the process. There are processes where installing extra sensors is not indicated, (e.g. the economical impact on electronic devices, insufficient space or extra weight in aircrafts). However, the presence of a non-diagnosed fault can cause serious problems in many industrial processes; to improve fault diagnosis the installation of extra sensors is fully recommended. The analysis by which it is determined that which sensors are needed to achieve certain specifications is called sensor placement analysis. There are some research areas where the sensor placement problem has been widely studied (e.g. observability in the control field) but this is not the case in diagnosis field, where there are only few isolated works present in the literature. However, this topic is gaining an increasing interest from last few years.

Till now, just few works had been dedicated to this problem [6] [7] [8] [9]. In [10] a part of the work proposed in this paper is already done which specifically focuses on analyzing which sensors should be installed in a process, to facilitate its design and at the same time improve the diagnosis capabilities of a model based diagnosis system. In [5] an alternate approach is shown to reduce number of links or communication between various nodes of a system. In the present work a novel algorithm is proposed...
to place the sensors in a system in such a manner that the partition of a system into various subsystems becomes easier, thus facilitating the implementation of a distributed fault diagnosis system.

The structure of the paper is the following: Section 2 describes the entire problem description. Section 3 describes the proposed methodology. Section 4 describes the implementation of the solution. Finally, in Section 5, conclusions are presented.

2. Problem Description

![Diagram](image)

**Figure 1.** Various blocks of decentralized/distributed fault diagnosis algorithm using ARRs

Figure 1 shows a block diagram of a decentralized/distributed fault diagnosis algorithm described in [2] [3]. The proposed algorithm in this paper is trying to find an optimal sensor placement solution such that when the set of selected ARRs are generated they have minimum coupling so that when applying the partitioning algorithm various subsystems are easily decoupled and partitioning of a system into various subsystem becomes easy. In the off-line phase, this fault diagnosis approach starts from obtaining the analytical redundancy relations (ARRs) using the system model and the set of available sensors. These ARRs are converted into a graph. This graph is further divided into various subgraphs using a partition algorithm. Each subgraph corresponds to a subsystem. From various subgraphs, different local fault signature matrices for various subsystems are obtained. Finally, in the on-line phase, using various local fault signature matrices, a set of diagnoser agents are created that allow the global diagnosis in a large scale system. For each subsystem, a local fault signature matrix is then obtained as next step. Using each local fault signature matrix, a local diagnoser is implemented by means of an agent. This agent is responsible for local diagnoses in a subsystem and communicating with other agents of others subsystems to perform global diagnosis. In order to demonstrate the applicability of the proposed approach, a case study based on the Barcelona drinking water network (DWN) was used. However, the main problem of this approach is that the set of sensors used for the generation of ARRs cannot be placed in such a manner that the ARRs generated are decoupled and thus ARRs produced by this approach are highly coupled and it is difficult to partition such highly coupled ARRs into various subsystems, to allow the distributed/decentralized implementation. Therefore, sensor placement in fault diagnosis is generally the first block or first step in a decentralized/distributed fault diagnosis algorithm so that the ARRs generated are decoupled and partition of ARRs into various subsystems becomes easier.
3. Proposed Algorithm
The proposed methodology for distributed fault diagnosis algorithm starts from discrete-time space state model. This model is the starting point for obtaining a set of analytical redundancy relations (ARRs) described below in section 3.1. The proposed methodology has 6 Blocks.

3.1 Block 1: Formation of set of all ARRs
ARRs are obtained from a model $M(z, x)$ by using ranking algorithm [13]. Let $z = \{z_1, z_2, z_3, \ldots, z_m\}$ be the set of the constraints which represent the system model and let $x = \{x_1, x_2, x_3, x_4, \ldots, x_n\}$ be the set of the variables which contains three subsets: let $k = u \cup y$ be the set of known variables, $u$ is the subset of input variables, $y$ is the subset of the output variables and $k$ is the subset of the unknown (non-measured) variables. The structure of the system model is described by the binary relation:

$$M : z \times x \rightarrow \{0, 1\}$$

where $(z_i, x_j) \rightarrow M(z_i, x_j)=1$ if $z_i$ applies to $x_j$ and $M(z_i, x_j)=0$, otherwise.

The unknown variables are replaced by known variables of the system model to obtain set of all ARRs or parity matrix $P$. From the set of all ARRs, a set of minimum coupled ARRs is selected by sensor placement algorithm which is described in detail below in section 3.2.

3.2 Block 1: Sensor Placement
Sensor placement algorithm is applied to set of all ARRs to obtain minimum coupled set of ARRs. The optimal sensor placement can be formulated as a Binary Integer Programming (BIP) problem and in literature it is described as Mixed Integer Programming. BIP optimization restricts the value of optimization variable to {0, 1}.The optimal sensor placement problem when solved, the sensor installation is represented by a binary vector, $q$, and each sensor $s_i \in S$ belongs to a binary element $q_i$ of $q$, with $q_i \in \{0, 1\}$. When $q_i$ equals 1, means the sensor must be installed but if $q_i$ equals 0, the sensor does not have to be installed.

The new work or constraints added in already existing sensor placement algorithm [4] [10] are the constraints to minimize coupling between ARRs which are described in equations (1) and (2) below.

3.2.1 Minimizing the coupling between ARRs
In order to facilitate the distributed implementation of the fault diagnosis systems, the sensors should be placed such that the coupling between ARRs is minimized. This is achieved by adding additional constraints that minimizes the number of common links between ARRs. First, a constraint that reduces the number of row links coupling is written in compact form as

$$
(W_{\omega j})^{i j} (0)_{i p} - I_{o i} (0)_{o a} \leq 0_{o i}
$$

An analog constraint could be added to minimize the row links coupling as follows

$$
(W_{\omega j})^T (0)_{j s} - I_{o j} \leq 0_{j s}
$$

Rest of the constraints of already existing sensor placement algorithm [4] [10] in the literature are described below.

3.2.2 Constraint Formulation for BIP
The proposed algorithm deals with fault detectability and isolability for process, sensors faults and also minimization of number of links in a system to ease partition of the system into various subsystems. To obtain this, it is assumed that the complete set $\Omega_S$ of all the analytical redundancy relations (ARRs) is available, when all the candidate sensors $S$ are installed in the system.

Let $n$ be the number of ARRs available in $\Omega_S = \{w_1, w_2, \ldots, w_n\}$, and let $k$ be the number of candidate sensors to be installed, i.e. $k = |S|$. A binary matrix $W = [w_{ij}]$ of size $n \times k$ be the set of ARRs (the row set) and sensors (the column set) is formed. Matrix $W$ is built from the set of ARRs sets as
\[ w_{ij} = \begin{cases} 1 & \text{if } e_{ij} \in W_j \\ 0 & \text{otherwise} \end{cases} \quad \text{(2)} \]

All \( w_i \in \Omega_S \) and all \( s_j \in S \) and \( w_{ij} = 1 \) means that ARR, depends on the sensor measurement \( s_j \). \( V \) is binary matrix which relates the process faults with the ARR sets. \( l \) is the number of process faults that have to be detected and isolated (i.e., \( l = |F_{DP}| \)). The binary matrix \( V = [v_{ij}] \) of size \( n \times l \) relates process faults (the column set) and the set of ARRs (the row set),

\[ v_{ij} = \begin{cases} 1 & \text{if } e_{ij} \in W_j \\ 0 & \text{otherwise} \end{cases} \quad \text{(3)} \]

The matrix \( V \) is known as the fault signature matrix (FSM). FSM includes both process and sensor faults and is represented as \((V, W)\). The first \( l \) columns represent process faults and the remaining \( k \) columns represent sensor faults.

### 3.2.3 AAR set selector

The set of all ARRs \( \Omega_S \) cannot be used to diagnose when a sensor is selected for installation. \( P \) is the ARR set selector vector, it is \( n \) elements binary vector (one for each ARR set) that indicates whether the ARR set is valid, for a candidate sensor set \( S \subseteq S \), i.e.

\[ \rho_i = 1 \quad \text{---the ARR set } W_i \text{ is valid} \]

\[ \rho_i = 0 \quad \text{---the ARR set } W_i \text{ is not valid} \quad \text{(4)} \]

\( w_i \in \Omega_S \) be an ARR set and \( s_j \in S \) be any of the sensors that depends on this MSO set \( w_i \) (i.e., \( e_{s_j} \in W_i \)). The ARR set is not valid till any of its corresponding sensors is not installed (i.e., \( q_j = 0 \)). Otherwise, the ARR set is valid. \( \rho_i \) is computed as

\[ \rho_i = \sum_{j=1}^{k} [w_{ij} q_j + (1 - w_{ij})] \quad \text{(5)} \]

If \( \rho_i = 1 \) the ARR set is valid and the ARR set is not valid if \( \rho_i = 0 \). Now, \( \lambda_i \) a binary variable is introduced in the inequality. \( \lambda_i \) is a dummy variable in the optimization problem and it is zero when \( \lambda_i \) is a non-valid ARR set. So,

\[ (-W \ kI_n) \begin{pmatrix} q \\ \lambda \end{pmatrix} \leq \beta \quad \text{(6)} \]

Where \( \lambda = (\lambda_1, ..., \lambda_n)^T \) is the set of linear ARR set selectors and \( \beta = (\beta_1, ..., \beta_n)^T \) is a vector of independent coefficients.

### 3.2.4 Process and sensor fault detectability

The constraint for process fault detectability written in compact form using the linear MSO set selector as

\[ \left(0_{1 \times k} - V^T \right) \begin{pmatrix} q \\ \lambda \end{pmatrix} \leq -1_{l \times 1} \quad \text{(7)} \]

The constraint for sensor fault detectability written in compact form using the linear ARR set selector as
3.2.5 Isolability between process and sensor faults

Let matrix $V_{ff} = [v_{ff_{im}}]$ be an $n \times C_{n}^{2}$ matrix such that

$$v_{ff_{im}} = |t_{ij1} - t_{ij2}| \quad \forall j_1, j_2 \in \{1, \ldots, l\} : j_1 < j_2$$

Where $m$ indexes, in alphabetic order, the set of pairs $(j_1, j_2)$

The constraint for Isolability between Process Faults written in compact form as

$$\begin{pmatrix} 0_{C_{n}^{2} - V_{ff}^{T}} \end{pmatrix} \begin{pmatrix} q \end{pmatrix} \leq -1_{C_{n}^{2} \times 1}$$

(10)

The constraint for isolability between process and sensor faults written in compact form as

$$\begin{pmatrix} G_{2} - V_{fs_{T}} \end{pmatrix} \begin{pmatrix} q \end{pmatrix} \leq 0_{l \times k}$$

(11)

Where $G_{2}$ is an $(l \cdot k) \times k$ matrix, given as

$$G_{2} = (I_{k} I_{k} \ldots I_{k})^{T}$$

(12)

and $V_{fs} = [V_{fs_{ip}}]$ is an $n \times (l \cdot k)$ matrix such that

$$V_{fs_{ip}} = (v_{ij1} - w_{ij2})^2 \quad \forall j_1 \in \{1, \ldots, l\}, \forall j_2 \in \{1, \ldots, k\}$$

(13)

Let the matrix $V_{ss} = [V_{ss_{it}}]$ of size $n \times C_{2}^{k}$ be

$$V_{ss_{it}} = (W_{g1} - W_{g2})^2 \quad \forall i, j_2 \in \{1, \ldots, k\} : j_1 < j_2$$

(14)

The constraint for isolability between sensor faults written in compact form as

$$\begin{pmatrix} G_{3} - V_{ss} \end{pmatrix}^{T} \begin{pmatrix} q \end{pmatrix} \leq 1_{C_{2}^{k} \times 1}$$

(15)

the matrix $G_{3}$ is a $C_{2}^{k} \times k$ matrix and built as

$$G_{3} = \begin{pmatrix} 1 & I_{k-1} \\ \\
1 & 0 & 1 \\ \\
\vdots & \vdots & \ddots & I_{k-2} \\ \\
0 & 1 & \ddots & 0 \\ \\
0 & \cdots & 0 & 1 & 1 \end{pmatrix}$$

(16)

3.2.6 Formulation of final optimization problem

The optimal sensor placement for fault detection, isolation and minimization of links for better partition is formulated as
\[
\min_{q, \lambda} \left( c^T O I_X \lambda \right)
\]
Subject to:

\[
\begin{pmatrix}
-W & kI_n \\
O_{i,j}^k & -V^T \\
I & -W^T \\
O_{c_{i,j}}^k & -V^T \\
G_i & -V^T \\
W & 0 \\
W^T & 0
\end{pmatrix}
\begin{pmatrix}
q \\
\lambda
\end{pmatrix}
\leq
\begin{pmatrix}
\beta \\
-1_{i,1} \\
0_{i,1} \\
-1_{c_{i,j},1} \\
0_{c_{i,j},1} \\
1_{i,1} \\
0_{i,1}
\end{pmatrix}
\]

(17)

Where \( q, \lambda \) have binary values while rows and columns have integer values.

After applying sensor placement algorithm, a set of minimum coupled set of ARR's are obtained. The ARR graph is obtained from the set of minimum coupled ARR's. Method of ARR graph generation is described in section 3.3 described below.

### 3.3 Block 3: ARR graph formation

Graph is generally defined as an abstract representation of a group of objects from a collection, where few pairs of objects are joint by links. The elements which are interconnected are typically called vertices while the connected links are called edges. If any 1 or -1 present in rows of parity matrix \( P \) makes that particular row a vertex of the graph and all the 1 or -1 present at same location of two rows is connected edge between the vertexes. From ARR matrix \( P \), the vertex and edges of graph are obtained and finally graph \( G(V, E) \), where \( V \) denotes the set of vertices, \( E \) is the set of edges. The graph \( G(V, E) \) can be represented in form of incidence matrix denoted as \( I_M \), which is defined such that

\[
I_{M_{ij}} = \begin{cases} 
-1 & \text{if the edge } e_j \text{ leaves vertex } v_i \\
1 & \text{if the edge } e_j \text{ enters vertex } v_i \\
0 & \text{otherwise}
\end{cases}
\]

This matrix has dimensions \( n_e \times n_v \), where \( n_v \) corresponds with the total number of vertices and \( n_e \) denotes the total number of edge. The graph obtained from the set of minimum coupled ARR's is further divided into various subgraphs or subsystems using partition algorithm defined below in section 3.4.

### 3.4 Block 4: Partition of ARR graph

The first step to implement partition algorithm [16] is to find the strongly connected vertices. A strongly connected vertex is the one which has maximum number of edges. This vertex will be the basis for forming the first subsystem being the core of the first subsystem. Second subsystem is formed by second strongly connected vertex. The important condition is that no two subgraphs can have same vertex but same edge can be shared. Together all the subgraphs must contain all the vertices of a system, that is, no vertex must be left. Every vertex must be part of any one subsystem and the subsystem should be least connected. The maximum weight \( \omega \) for each vertex is equal to number of edges each vertex have. The heaviest vertex is the vertex which has maximum number of edges, the heaviest vertex forms the first subgraph and the centre of the first subgraph \( G_{i1} \) is defined. Those vertices which are connected to this heaviest vertex are included in first subsystem. The important condition is that no two subgraphs can have same vertex but same edge can be shared.

1: \( I_M \rightarrow \) System topology
2: \( G(V, E) \rightarrow I_M \)
3: \( \text{for } j = 1 \text{ to } \emptyset \) do
4: Compute \( \omega_j \)
5: end for
% Partitioning
6: \( V_r \leftarrow V = 1 \)
7: repeat
8: Find \( v \in V_r \) with maximum \( \omega \)
9: \( V_i \leftarrow v \) and all its neighbour vertices
10: \( V_r = V - \{ \bigcup_{h=1}^{n} V_h \} \)
11: \( i = i + 1 \)
12: until \( V_r = \emptyset \)

After obtaining various subsystems, fault signature matrices or local diagnosers are obtained by the method defined below in section 3.5.

3.5 Block 5: Fault signature matrices of local diagnosers

The subsystem or sets of minimum coupled ARRs are represented as
\[
R = \{ r_i | r_i = \Psi_i(y_k, u_k, \theta_k), i = 1, \ldots, n_r \}
\]
(19)
\( \psi_i \) is the mathematical expression for ARR sets and \( n_r \) is the ARR sets number obtained. Fault diagnosis is done by identifying the set of consistent ARR sets
\[
R_0 = \{ r_i | r_i = \Psi_i(y_k, u_k, \theta_k), \neq 0, i = 1, \ldots, n_r \}
\]
(20)
and inconsistent ARR sets
\[
R_1 = \{ r_i | r_i = \Psi_i(y_k, u_k, \theta_k), \neq 0, i = 1, \ldots, n_r \}
\]
(21)
when some inconsistency in (19) at time instant \( k \) is detected, the process of fault isolation starts by obtaining the observed fault signature, where each single fault signal indicator is defined as follows:
\[
\phi_i(k) = \begin{cases} 
0 & \text{if } r_i(k) \in R_0, \\
1 & \text{if } r_i(k) \in R_1 
\end{cases}
\]
(22)
Fault isolation is the binary relation between the considered fault hypothesis set \{ \( f_1(k), f_2(k) \), \ldots, \( f_{nf}(k) \) \} and the fault signal indicators \( \phi_i(k) \), stored in the Fault Signature Matrix \( F \). Till Block 5 the methodology is in offline mode. Block 6 or distributed fault diagnosis works in online mode and the method is described below in section 3.6.

3.6 Block 6: Distributed fault diagnosis

A distributed system compose of a set of agents \( \mathcal{A} = \{ A_1, \ldots, A_n \} \) and each agent includes a diagnostic system. A local diagnosis in such distributed systems is a diagnosis that is determined by violated residuals in one agent, while a global diagnosis is determined by all violated residuals in all agents. The violated residuals are generated from responded diagnostic tests. A set of local diagnoses in agent \( A_i \) is denoted by \( \mathcal{D}^{A_i} \) and a set of global diagnoses is denoted by \( \mathcal{D} \). The set of global diagnoses can be computed from the sets of violated residuals in all agents. The global diagnoses can also be computed by extracting the diagnoses in the set resulting from a merge of all sets of local diagnoses. This idea of computing the global diagnoses from the sets of local diagnoses will be used in the algorithm which computes the set of diagnoses in each agent. The diagnosis in one agent should only include the variables that the agent uses. To be able to decide which variable that an agent uses, the variables are divided into internal variables \( \mathcal{U} \subseteq \mathcal{C} \) and shared variables \( \mathcal{V} \subseteq \mathcal{C} \) (where \( \mathcal{C} \) is set of entire variables including both internal as well as shared variable). A internal variable is only used by one agent, while a shared variable is used by two or more agents. The set of internal variables is therefore further divided into different sets belonging to different agents, where the set \( \mathcal{U}^{A_i} \subseteq \mathcal{U} \) is used by agent \( A_i \).
The outline of the algorithm is that each agent first computes its set of local diagnoses, and then
transmits, to all other agents, the local diagnoses that include variables used by other agents. After this,
each agent merges the received sets with its own set of local diagnoses which results in the set of
diagnoses.

The transmitting algorithm is described in detail in algorithm 1, and the receiving and merging
algorithm is described in algorithm 2. Finally, the main algorithm is described in algorithm 3.

Algorithm 1 – Transmit ($\mathcal{D}^i$). Transmit subsets of local diagnoses.

Input: A set of local diagnoses $\mathcal{D}^i$
Output: Set of tuples $TX^i$ representing the diagnoses in $\mathcal{D}^i$ that should be transmitted.
1: $TX^i := \{ \mathcal{D} \in \mathcal{D}^i : \mathcal{D} \cap \mathcal{V} \neq \emptyset \}$
2: $TX^i := \emptyset$
3: for each $D \in TX^i$ do
4: $TX^i := TX^i \cup \{ D \}$
5: end for

It is seen that a local diagnosis in one agent is of interest for the other agents if it includes variables
used by some other agent. The local diagnoses including such variables should be transmitted to the
other agents. Before the diagnoses are transmitted, the internal variables can be removed since these
are not used by any other agent.

Algorithm 1 performs the steps described above. Row 1 decides which local diagnoses that include
variables used by other agents, resulting in the set $TX^i$. Rows 2–7 construct a tuple for each local
diagnosis $\mathcal{D} = \mathcal{U} \cup \mathcal{V}$ in the set $TX^i$, where the set $\mathcal{U}$ is the internal variable, $\mathcal{V}$ is the shared variable
included in the local diagnosis. Each tuple in the set $TX^i$ includes a set consisting of $\mathcal{V}$ which is taken
directly from the local diagnosis which depends on any of their moved local variables in $\mathcal{U}$.

Algorithm 2 – Receive ($\{ TX^1, \ldots, TX^n \}$). Compute the diagnoses in agent $A_i$.

Input: For each agent $A_j$ except $A_i$, a received set $TX^j$ resulting from the evaluation of Transmit($\cdot$).
The set of local diagnoses $\mathcal{D}^i$
Output: The set of diagnoses $D^i_{\mathcal{A}}$.
1: for each $A_j$ except $A_i$ do
2: $RX^j := \emptyset$
3: for each $D \in TX^j$ do
4: $RX^j := RX^j \cup \{ D \}$
5: end for
6: end for
7: $RX^j := \{ D, D \in \mathcal{D}^j \}$
8: $D^i_{\mathcal{A}} := \text{CrossProductTuple}(\{ RX^1, \ldots, RX^n \})$
Algorithm 2 performs the steps described above. Rows 1–7 transform each set of transmitted tuples, such as $\mathcal{T}^k_i$ transmitted by agent $A_i$, into sets of received tuples, such as $\mathcal{R}^k_i$. The received tuple $\{v\}$ is thereafter constructed and stored in the set of tuples $\mathcal{R}^k_i$. Row 8 transforms the local diagnoses in agent $A_i$ to the same format as the received tuples. In rows 8 in the algorithm, the sets $\mathcal{R}^k_1, \ldots, \mathcal{R}^k_n$ are merged. The function CrossProductTuple for a set $\mathcal{M}$ of sets of tuples performs a normal cross product with respect to the sets included in the tuples.

Algorithm 3 – Main. Computation of the set of diagnoses in each agent.

**Input:** The set of violated residuals $\prod^k_i$ in each agent $A_i$.

**Output:** The set diagnoses $D^k_i$ in each agent $A_i$.

1: for each agent $A_i$ do $D^k_i := \text{Diagnoses}(\prod^k_i)$ in $A_i$
2: for each Agent $A_i$ do
3: compute $\mathcal{T}^k_i := \text{Transmit}(D^k_i)$ in agent $A_i$
4: broadcast $\mathcal{T}^k_i$ on the network
5: end for
6: for each Agent $A_i$ do
7: receive $\mathcal{T}^k_i$ in $A_i$ from all agents $A_j$ except $A_i$
8: compute $D^k_i := \text{Receive}((\mathcal{T}^k_1, \ldots, \mathcal{T}^k_n \setminus \mathcal{T}^k_i))$ in $A_i$
9: end for

Algorithm 3 used to compute the set diagnoses in each agent. For each agent, the algorithm computes the set of minimal local diagnoses $D^k_i$ in row 1 using a minimal hitting set algorithm. The algorithm then evaluate rows 2–5 which include calls to Algorithm 1, where each call has a set of local diagnoses as input and gives a set of transmitted diagnoses $\mathcal{T}^k_i$ as output. Finally, rows 6–9 are evaluated which includes calls to Algorithm 2, where each call has the set of transmitted sets $\mathcal{T}^k_1, \ldots, \mathcal{T}^k_n$ except $\mathcal{T}^k_i$ as input. After that Algorithm 3 has been evaluated, each agent has a unique set of minimal condensed diagnoses $D^k_i$.

4. Implementation of Algorithm

The proposed algorithm is implemented on real Barcelona water network shown in Figure 2 below. The Barcelona DWN, managed by Aguas de Barcelona, S.A. (AGBAR), supplies drinking water to Barcelona city and also to the metropolitan area. The sources of water are the Ter and Llobregat rivers, which are regulated at their head by some dams with total capacity of 600 cubic hectometres. Currently, there are four drinking water treatment plants (WTP): the Abrera and Sant Joan Despí plants, which extract water from the Llobregat river, the Cardedeu plant, which extracts water from the Ter river, and the Beso`s plant, which treats the underground flows from the aquifer of the Beso`s river. Several underground sources (wells) are also there that can provide water through pumping stations. Different water sources currently provide a flow of around 7 m$^3$/s. The water flow from each source is limited and with different water prices depending on legal extraction canons and water treatments. The Barcelona DWN consists of two layers. The upper layer links the water treatment plants with the reservoirs distributed all over the city and named as transport network. The lower layer is sectorised in sub networks and named distribution network. Each sub network generally links a reservoir with each consumer. Focus of this paper is on the transport network. Each sub network of the distribution network is modelled as a demand sector. The demand of each sector can be predicted by
using a time-series model and is characterised by a demand pattern. The control system of the transport network is also consists of two layers. The upper layer establishes the set-points of the regulatory controllers at the lower layer and also is in charge of the global control of the network. The supervisory layer controller is of MPC type and regulatory controllers are of proportional–integral–derivative (PID) type. Regulatory controllers hide the network non-linear behaviour to the supervisory controller and thus allow the MPC supervisory controller to use a control oriented linear model.

This modelling methodology has been applied to the Barcelona DWN aggregate network in Figure 2. From this figure, it can be seen that the network comprises 17 tanks (state variables), 61 actuators (26 pumping stations and 35 valves), 11 nodes and 25 main sectors of water demand (model disturbances). The model has been simulated and compared against real behaviour assessing its validity. The detailed information about physical parameters and other system values are reported in [15].

The following equation or set of all ARRs describes the aggregated Barcelona water network used for real case study shown in figure 2.

**Tank 1:**  
\[ x_1(k + 1) = x_1(k) + \Delta_t (u_{19}(k) - d_8(k)) \]  
**Tank 2:**  
\[ x_2(k + 1) = x_2(k) + \Delta_t (u_{12}(k) + u_{15}(k) - d_{10}(k)) \]  
**Tank 3:**  
\[ x_3(k + 1) = x_3(k) + \Delta_t (u_{42}(k) + u_{35}(k) + u_{52}(k) + u_9(k) + u_{13}(k) + u_{23}(k) - u_6(k) - u_{15}(k) - u_{16}(k) - u_{47}(k) - d_{12}(k)) \]  
**Tank 4:**  
\[ x_4(k + 1) = x_4(k) + \Delta_t (u_{4}(k) + u_{15}(k) + u_{16}(k) - u_{24}(k)) \]  
**Tank 5:**  
\[ x_5(k + 1) = x_5(k) + \Delta_t (u_{2}(k) + u_{20}(k) - d_{11}(k)) \]  
**Tank 6:**  
\[ x_6(k + 1) = x_6(k) + \Delta_t (u_{16}(k) - u_{53}(k) - d_{20}(k)) \]  
**Tank 7:**  
\[ x_7(k + 1) = x_7(k) + \Delta_t (u_{47}(k) + u_s(k) + u_{24}(k) + u_6(k) - u_{13}(k) - u_{14}(k) - u_{59}(k) - u_{61}(k) - u_{57}(k) - d_4(k)) \]  
**Tank 8:**  
\[ x_8(k + 1) = x_8(k) + \Delta_t (u_{3}(k) + u_5(k) + u_{12}(k) - u_{50}(k) - d_d(k)) \]  
**Tank 9:**  
\[ x_9(k + 1) = x_9(k) + \Delta_t (u_{30}(k) + u_4(k) + u_{50}(k) + u_{17}(k) - u_{12}(k) - u_7(k) - u_{46}(k) - d_{25}(k)) \]  
**Tank 10:**  
\[ x_{10}(k + 1) = x_{10}(k) + \Delta_t (u_{40}(k) + u_{50}(k) + u_{55}(k) - u_{18}(k) - d_{23}(k)) \]
The proposed distributed fault diagnosis algorithm starts from discrete-time space state model

\[ x(k+1) = Ax(k) + Bu(k) + B_d d(k) \]

\[ y(k) = C x(k) \]  

(23)

where \( A \in \mathbb{R}^{m \times m} \), \( B \in \mathbb{R}^{m \times n} \), \( C \in \mathbb{R}^{p \times m} \) are the state space matrices and \( B_d \in \mathbb{R}^{n \times q} \) is the disturbance known, \( x \in \mathbb{R}^n \) is the state vector corresponding to the volume of deposits, \( u \in \mathbb{R}^m \) is the vector of input variables, \( d \in \mathbb{R}^q \) corresponds the vector of known disturbances, in this case are the water demands, \( y \in \mathbb{R}^p \) is the vector of outputs.
After obtaining model from the sets of equations or set of all ARRs, sensor placement algorithm is applied on the given model. W matrix represents sensor faults while V matrix represents process faults. W and V matrix is obtained using equations C1 upto C28 described above. W matrix has constraints (C1, C2...etc) or ARRs in rows and variables in column (x1, u1, d1...etc), total W matrix has 28 rows and 98 columns. If a variable present in an equation or row, that particular column has 1 otherwise the column contains zero, similarly V matrix has 28 rows and 4 columns. V matrix has constraints (C1, C2...etc) or ARRs in rows and process or main supply variables in columns (u25, u70 (u28, u29), u30).

Sensor placement algorithm is applied on the W and V matrix (Barcelona water network) described above, the algorithm chosen 4 ARRs C4, C11, C18, C28 as shown below in Table1 from the set of all ARRs. The chosen system or set of ARRs has minimum coupling. The system obtained (shown in Table 1) is first converted into graph shown in Figure 3 and this graph is partitioned into different subsystem or subgraph using partition algorithm. The system is divided into 3 subsystems. The first subsystem consists of ARRs C4, C11, the second system consists of ARR C18 and the third subsystem consists of ARR C28. The details of the 3 subsystems are shown in Fig 2, Fig 3 and Table 2.

|   | X4 | X11 | U10 | U19 | U21 | U24 | U26 | U27 | U30 | U31 | U32 | U34 | U35 | U36 | U40 | U41 | U42 | U49 | U70 | d13 |
|---|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|C4|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|
|C11|0 1|0 1|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|1 0|
|C18|0 0|0 1|0 1|0 1|1 0|1 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|
|C28|0 0|0 0|0 0|0 0|0 0|0 1|0 1|0 1|0 1|0 1|0 1|0 1|0 1|0 1|0 1|0 1|0 1|0 1|0 1|

From the 3 subsystems, 3 fault signature matrices are created shown in Table 3, 4, 5. These fault signature matrices are represented by 3 agents A1, A2, A3. Agents A1, A2, A3 communicate with each other to detect and isolate a fault or faults. A scenario is explained below

Till now all steps performed are offline, the fault detection is done online. A scenario is taken that there are 3 subsystems (3, 4, 5) represented by 3 agents A3, A4, A5. Suppose in the subsystem 3, MSO 10 and variable 51 is faulty or violated and similarly in subsystem 4, MSO 24 and variable 51 is faulty or violated and in subsystem 5, no MSO is faulty or violated. So how the 3 agents will communicate with each other to diagnose and isolate the fault using concept of distributed fault diagnosis (local diagnosis with minimum global diagnosis) following traditional

FDI approach, this entire process is explained below in point wise manner. Everything happening together and parallel shown in Fig 4.4 to 4.8

1. First all Agents A3, A4 and A5 do local diagnosis (D^hi) according to algorithm 1. After doing local diagnosis agent A3 finds out that MSO 10 is faulty and the single fault can occur in shared variable 51 or internal variable 56. Similarly A4 finds out that MSO 24 is faulty and single fault can occur in shared variables 33 38 or 51 and A5 finds out that no MSO is faulty after local diagnoses. A3 sends a communication message (D^hi) containing shared variable number
51 as data to agent $A_4$ and $A_5$. Similarly at the same time $A_4$ will send a message containing
shared variable numbers 33 38 51 to $A_3$ and $A_5$ and $A_5$ will send a message containing 00
indicating no fault to $A_3$ and $A_4$. Entire process is explained with help of arrow diagram below.

1. for Agent $A_3$ do
Checks residual value of each $MS^O$ of
Agent $A_3$ whether 0 or not according to (3.11)
end for

2. $A_3 (\mathcal{R}^A_1 = 51, 56) A_4, A_5$

3. for Agent $A_4$ do
Checks residual value of each $MS^O$ of
Agent $A_4$ whether 0 or not according to (3.11)
end for

4. $A_4 (\mathcal{R}^A_2 = 33, 38, 51) A_3, A_5$

5. for Agent $A_5$ do
Checks residual value of each $MS^O$ of
Agent $A_5$ whether 0 or not according to (3.11)
end for

6. $A_5 (\mathcal{R}^A_3 = 00) A_3, A_4$

2. In reply of message from $A_3$, $A_4$ and $A_5$ will first perform local diagnoses ($D^H$) according to
algorithm 2. $A_3$ will receive message ($\mathcal{R}^H_1$) containing data 00 from $A_5$, indicating that there
is no error or fault and will receive message containing data (33, 38, 51) from $A_4$. $A_4$ will
receive message containing data 00 from $A_5$ and will receive message containing data 51 from
$A_3$.

3. After each agent receives the information, on the basis of this information the agents $A_3$, $A_4$
and $A_5$ will detect and isolate the variable in which the single fault occurred ($D^V$) according
to algorithm 3. In this scenario the agent $A_5$ detect that single fault has occur in shared variable
51 and not in internal variable 56 and agent $A_4$ detect that single fault has occur in shared
variable 51 and not in shared variables 33 38.
Figure 3. Showing three subsystems obtained

Table 2. Three subsystems obtained

| Subsystem (ARRs) | Color   |
|-----------------|---------|
| C4, C11         | Yellow  |
| C18             | Red     |
| C28             | Blue    |

Table 3. Local fault signature matrix of subsystem 1 diagnosed by agent $A_1$

|       | X4 | X11 | U10 | U19 | U21 | U22 | U24 | U26 | U27 | U30 | U31 | U32 | U34 | U35 | U36 | U37 | U40 | U41 | U42 | U49 | U70 | D13 |
|-------|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| C4    | 1  | 0   | 0   | 0   | 0   | 1   | 0   | 0   | 0   | 0   | 1   | 0   | 0   | 0   | 0   | 1   | 0   | 0   | 0   | 0   | 0   | 0   |
| C11   | 0  | 1   | 0   | 0   | 1   | 0   | 1   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 1   | 0   |

Table 4. Local fault signature matrix of subsystem 2 diagnosed by agent $A_2$

|       | X4 | X11 | U10 | U19 | U21 | U22 | U24 | U26 | U27 | U30 | U31 | U32 | U34 | U35 | U36 | U37 | U40 | U41 | U42 | U49 | U70 | D13 |
|-------|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| C18   | 0  | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 1   | 0   | 1   | 1   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

Table 5. Local fault signature matrix of subsystem 2 diagnosed by agent $A_3$

|       | X4 | X11 | U10 | U19 | U21 | U22 | U24 | U26 | U27 | U30 | U31 | U32 | U34 | U35 | U36 | U37 | U40 | U41 | U42 | U49 | U70 | D13 |
|-------|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| C28   | 0  | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 1   | 0   | 0   | 1   | 1   | 1   | 1   | 1   | 1   | 1   | 1   | 0   | 1   |
5. Conclusion
In this paper, a methodology distributed fault diagnosis is proposed. The objective of this methodology is to place the sensor to measure the unknown variables in a system in such a manner that detection of sensor faults as well as process faults become easier and also the partition of a system into various subsystems become more easy and simplified by reducing number of links within a system. This algorithm also reduces or minimized the number of sensors to be used or install thus reducing overall cost. The algorithm has been satisfactorily implemented on Barcelona water network.

References
[1] V. Vekatasubramanian, R. Rengaswamy, K. Yin, and S. Kavuri, 2003. A review of process fault detection and diagnosis part I: Quantitative model-based methods. Computer and Chemical Engineering, 27:293–311.
[2] Vikas Gupta, Vicenç Puig, 2016. “Decentralized Fault Diagnosis using Analytical Redundancy Relations: Application to a Water Distribution Network,” in proceedings of 15th European Control Conference ECC. Aalborg, Denmark.
[3] Vikas Gupta, Vicenç Puig, 2016. “Distributed Fault Diagnosis using Analytical Redundancy Relations: Application to a Water Distribution Network,” in proceedings of 3rd International Conference on Control and Fault-Tolerant Systems. Barcelona, Spain.
[4] Vikas Gupta, Vicenç Puig, 2016. “Sensor placement in Distributed Network,” in proceedings of 3rd International Conference on Control and Fault-Tolerant Systems, Barcelona, Spain.
[5] H. Khorasgani, G. Biswas and D. Jung, 2015. “Minimal Structurally Overdetermined Sets Selection for Distributed Fault Detection,” Proceedings of the 26th International Workshop on Principles of Diagnosis, 75-82.
[6] R. Raghuraj, M. Bhushan, and R. Rengaswamy, 1999. “Locating sensors in complex chemical plants based on fault diagnostic observability criteria,” AIChE J., 45(2):310–322.
[7] M. Basseville, A. Benveniste, G. Moustakides, and A. Rougee, 1987. “Optimal sensor location for detecting changes in dynamical behavior,” IEEE Transactions on Automatic Control, 32(12):1067 – 1075.
[8] M. Bagajewicz, 2000. “Design and Upgrade of Process Plant Instrumentation,” Technomic Publishers, Lancaster, PA.
[9] L. Travé-Massuyès, T. Escobet, and X. Olive, 2006. “Diagnosability analysis based on component supported analytical redundancy relations,” IEEE Transactions on Systems, Man, and Cybernetics-Part A, 36(6):1146–1160.
[10] A. Rosich, 2011. “Sensor Placement for Fault Diagnosis based on Structural Models: Application to a Fuel Cell Stack System,” PhD Thesis, UPC.
[11] A. Rosich, R. Sarrate, and F. Nejjar, 2009. “Optimal sensor placement for FDI using binary integer programming,” International Workshop on Principles of Diagnosis.
[12] H. Khorasgani, D. E. Jung, G. Biswas, E. Frisk, and M. Krysander, 2014. “Off-line robust residual selection using sensitivity analysis,” International Workshop on Principles of Diagnosis (DX-14), Graz, Austria.
[13] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, 2016. “Diagnosis and Fault-Tolerant Control,” Springer, 3rd Edition.
[14] Laurence A. Wolsey, 1998. “Integer programming,” volume 42, Wiley, New York.
[15] Fambrini, V. Ocampo-Martinez, C. 2009. ‘Modelling and decentralized model predictive control of drinking water networks’. Technical report IRI-TR-04-09, Institut de Robòtica i Informàtica Industrial (CSIC-UPC).
[16] J. Biteus, EC. Ocampo-Martinez, S. Bovo, and V. Puig, 2011. “Partitioning approach oriented to the decentralised predictive control of large-scale systems,” Journal of Process Control, 21(5):775–786.
[17] V. Puig, C. Ocampo-Martinez, 2015. “Decentralised fault diagnosis of large-scale systems: Application to water transport networks,” 26th International Workshop on Principles of Diagnosis.