2013 European Control Conference (ECC)
July 17-19, 2013, Zürich, Switzerland.

A Parallel Algorithm for Optimum Monitoring Network Design in Parameter Estimation of Distributed Systems

Dariusz Uciński and Przemysław Baranowski

Abstract— **The design of a network of observation nodes in a spatial domain is addressed. The observations are to be used to estimate unknown parameters of a distributed parameter system. Given a finite number of possible sites at which to locate a sensor, the problem is formulated as the selection of the gauged sites so as to minimize a convex criterion defined on the Fisher information matrix associated with the estimated parameters. The search for an optimal solution to this binary optimization problem is performed through solving a relaxed problem in which a constrained discrete probability distribution on the set of all allowable sites is sought. The main contribution here consists in properly parallelizing this solution using the parallel variable distribution approach. As a result, each processor minimizes a convex function subject to linear constraints through the use of a simplicial decomposition algorithm. The resulting individual solutions are then synchronized by finding their optimal convex combination.**

I. INTRODUCTION

One of the crucial design problems in parameter estimation of Distributed Parameter Systems (DPSs), i.e., systems modelled by partial differential equations (PDEs), is the issue of where to locate the measurement sensors. Over the past years, this question has stimulated laborious research on the development of strategies for efficient sensor placement [1]–[6]. Nevertheless, although the need for systematic methods is widely recognized, most techniques communicated by various authors concerned with applications usually rely on exhaustive search over a predefined finite set of candidate locations and the combinatorial nature of the design problem is taken into account very occasionally, cf. [7], [8]. Obviously, this approach, which is affordable for a relatively small number of possible locations, soon becomes useless as the number of possible location candidates is increased.

To a great extent, a barrier to widespread use of existing optimal sensor location techniques is the complexity and large scale of the attendant computations. This issue becomes of paramount importance in case the design is supposed to be performed *on-line*. The works [9]–[11] were intended as attempts to demonstrate that these computations can be parallelized with relative ease and even for relatively complex and large-scale cases solutions can be obtained in reasonable times on a cluster of low-cost PCs using the Message Passing Interface (MPI) library.

D. Uciński is with the Institute of Control and Computation Engineering, University of Zielona Góra, ul. Podgórna 50, 65–246 Zielona Góra, Poland d.ucinski@issi.uz.zgora.pl

P. Baranowski is with Computer Centre, University of Zielona Góra, ul. Podgórna 50, 65–246 Zielona Góra, Poland p.baranowski@ck.uz.zgora.pl

This paper extends that approach by addressing the design of a network of observation locations in a spatial domain that will be used to estimate unknown parameters of a possibly nonlinear DPS. We consider a setting where we are given a finite number of possible sites at which to locate a sensor, but cost constraints allow only some proper subset of them to be selected. Consequently, we formulate this problem as the selection of the gauged sites so as to minimize some design criterion defined on the Fisher information matrix associated with the estimated parameters.

The search for the optimal solution is attacked by relaxing the original combinatorial problem, which then boils down to seeking an optimal discrete probability distribution on the set of allowable sites. Thus each site is assigned a nonnegative weight which quantifies its potential contribution to the expected accuracy of the parameter estimates. Then the parallel variable distribution (PVD) framework is used to solve the relaxed problem and the sensors are located at sites corresponding to the computed weights which have largest values.

The main contribution of the paper consists in substantially extending the scale of solvable combinatorial sensor location problems by exploiting parallelism which is more and more widespread in modern hardware. In the PVD scheme, the variables are distributed among parallel processors with each processor having the primary responsibility for updating its block of variables while allowing the remaining variables which are associated with other processors to change in a restricted fashion along some easily computable directions. Thus each processor minimizes a convex function subject to linear constraints and the resulting individual solutions are then synchronized by finding their optimal convex combination. This scheme is repeated over and over until a sufficiently accurate approximation to the sought global optimum is found. The local optimization tasks assigned to individual processors are solved via the simplicial decomposition algorithm which alternates between the column-generation problem (it boils down to a linear-programming task), and the restricted master problem which can be solved using standard convex optimization procedures. What is more, on the case of the D-optimality criterion, the latter problem can be addressed using an extremely simple multiplicative weight optimization algorithm widely used in optimum experimental design theory.

The use of the proposed approach is illustrated by an example involving sensor selection for a two-dimensional convective diffusion process. The implementation was made in a cluster of PCs based on the MPI paradigm.

978-3-952-41734-8/©2013 EUCA

II. COMBINATORIAL SENSOR LOCATION PROBLEM AND ITS RELAXATION

Consider a DPS defined on a time interval $T = (0, t_f]$ and in a spatial domain $\Omega \subset \mathbb{R}^2$ with a sufficiently smooth boundary Γ. We assume that it is modelled by the PDE

$$\frac{\partial y}{\partial t} = \mathcal{F}(x, t, y, \theta), \quad x \in \Omega,\ t \in T, \tag{1}$$

where $\mathcal{F}$ is a given operator including first and second-order spatial derivatives, $y = y(x,t)$ is the value of the scalar state variable at point $x \in \Omega$ and at time moment t. This description is complemented with appropriate boundary and initial conditions. Equation (1) is parametrized by a vector $\theta \in \mathbb{R}^r$ whose value is unknown, but is going to be estimated based on measurements from a number of available sensors to be located in Ω. The implicit dependence of y on θ will be reflected by the notation $y(x,t;\theta)$.

We consider a setting where we are given a finite number of possible sites $x^1, \ldots, x^m$ at which to locate a sensor, but cost constraints allow only some proper n-element subset of them to be selected. At each gauged site x^i the state variable y can be observed through a sensor which yields the observations of the form

$$z^i(t) = y(x^i, t; \theta) + \varepsilon(x^i, t), \quad t \in T, \tag{2}$$

where $z^i(t)$ is the sensor output and $\varepsilon(x^i, t)$ stands for measurement noise. We assume that ε is a realization of a stationary zero-mean Gaussian random field which is uncorrelated in both space and time. This means, among other things, that the measurements from different sensors are independent of one another.

Our aim is to select the gauged sites so as to maximize some measure of the expected estimation accuracy for the parameter vector θ. According to optimum experimental design theory [4], [12], [13], such measures are usually based on the following average Fisher Information Matrix (FIM), whose inverse constitutes (up to a constant multiplier) an approximation to the covariance matrix of the estimates:

$$M(w) = \sum_{i=1}^{m} w_i S_i, \tag{3}$$

where

$$S_i = \int_T g(x^i, t) g^{\mathsf{T}}(x^i, t)\, \mathrm{d}t, \tag{4}$$

$$g(x,t) = \left[\frac{\partial y(x,t;\theta)}{\partial \theta_1}, \quad \ldots, \quad \frac{\partial y(x,t;\theta)}{\partial \theta_r} \right]^{\mathsf{T}}_{\theta=\theta^0} \tag{5}$$

is the vector of the so-called sensitivity coefficients, θ^0 being a preliminary estimate of θ. Here the binary weights w_i are either $1/n$ or 0 depending on whether or not a sensor is located at x^i, respectively. Clearly, since there are n sensors, these weights sum up to unity.

Note that the matrices S_i, $i = 1, \ldots, m$ can be computed and stored in computer memory prior to solving the sensor location problem. Also observe that the table

$$\xi = \begin{Bmatrix} x^1, & x^2, & \ldots, & x^m \\ w_1, & w_2, & \ldots, & w_m \end{Bmatrix}, \tag{6}$$

which is called a *design*, can be interpreted as a discrete probability distribution which concentrates on a support set consisting of a finite number of x^i values.

Obviously, the binary nature of the above designs causes serious difficulties, as the resultant combinatorial problem is not amenable to solution by standard optimization techniques when m becomes large. A device which is commonly used in optimum experimental design to cope with this predicament is to extend the definition of the design. Adopting this line of reasoning, in the relaxed formulation considered in what follows, the feasible w_i's will be considered as any real numbers in the interval $[0, 1/n]$ which sum up to unity, and not necessarily as either 0 or $1/n$. 'Optimal' weights so computed can serve as indicators of spatial subdomains which furnish most valuable information on the estimated parameters. When returning to the original sensor location problem, n largest weight values can be used to indicate sites at which to locate sensors. (Observe that the constraint that the weights must not exceed $1/n$ guarantees that there are always at least n nonzero weights.)

Note that the weights can also be interpreted as required sensor precisions (a larger weight at a support point indicates the necessity of locating a more precise measurement device at that point).

In what follows, we thus seek designs, in which the optimal weights are

$$w^\star = \arg\min_{w \in W} Q(w), \tag{7}$$

where $Q(w) = \Psi[M(w)]$ for some design criterion Ψ being a convex function defined over the cone of positive-definite matrices, and the feasible domain W is

$$W = \Big\{ w \in \mathbb{R}^m : \sum_{i=1}^{m} w_i = 1, \quad 0 \le w_i \le \frac{1}{n},\ i = 1, \ldots, m \Big\}. \tag{8}$$

The introduction of an optimality criterion permits to cast the sensor location problem as an optimization problem. Several choices exist for this function [12], [14] and here we consider the most popular ones:

- The D-optimality criterion

$$\Psi(M) = \log\det(M^{-1}), \tag{9}$$

- The A-optimality criterion

$$\Psi(M) = \mathrm{tr}(M^{-1}). \tag{10}$$

A D-optimum design minimizes the volume of the confidence ellipsoid for the estimates. An A-optimum design suppresses the average variance of the estimates.

III. ALGORITHM BASED ON PARALLEL VARIABLE DISTRIBUTION

A. General Scheme

The proposed efficient parallel algorithm to determine a Ψ-optimum design $w^\star$ is based on application of the parallel

variable distribution (PVD) approach [15]. We assume that there are p processors which can perform parallel computational tasks and one processor which supervises these computations. Accordingly, we partition the decision variable $w \in \mathbb{R}^m$ into p blocks so that

$$w = (v_1, \ldots, v_p), \quad v_\ell \in \mathbb{R}^{m_\ell}, \quad \ell = 1, \ldots, p \tag{11}$$

with $\sum_{\ell=1}^{p} m_\ell = m$. Processor ℓ attempts to find (7), but it can freely update only its "primary" block component v_ℓ, while the remaining $p-1$ "secondary" block components are allowed to change in a restricted fashion only along some easily computable PVD-directions. These changes, in turn, are steered by the vector of coordinating variables $\mu_\ell \in \mathbb{R}^{p-1}$. The $p-1$ components of μ_ℓ represent in a condensed form all the remaining $m - m_\ell$ decision variables.

Each such processor may rightly be called a *slave* processor, as it is coordinated by a single *master* processor. The slaves solve local optimization problems with $m_\ell + p - 1$ decision variables (which can be substantially lower than m) and are then coordinated by the master processor which seeks to solve the global problem (7). This computational model is displayed as Algorithm 1.

Algorithm 1 Skeletal scheme of PVD.

1: (**Initialization**) Choose $w^0 \in W$ and set $k \leftarrow 0$.
2: **while** not (stopping_condition(w^k)) **do**
3: (**Variable distribution**) Given w^k, determine
4: a direction $d^k \in \mathbb{R}^m$ and partition both
5: the vectors as

$$w^k = (v_1^k, \ldots, v_p^k), \quad d^k = (d_1^k, \ldots, d_p^k),$$

6: where $v_\ell^k, d_\ell^k \in \mathbb{R}^{m_\ell}$, $\ell = 1, \ldots, p$.
7: (**Parallel computations**) For $\ell \in \{1, \ldots, p\}$,
8: Processor ℓ determines

$$\hat{v}_\ell^k \in \mathbb{R}^{m_\ell},$$
$$\hat{\mu}_\ell^k = (\hat{\mu}_{\ell,1}^k, \ldots, \hat{\mu}_{\ell,\ell-1}^k, \hat{\mu}_{\ell,\ell+1}^k, \ldots, \hat{\mu}_{\ell,p}^k) \in \mathbb{R}^{p-1}$$

9: which minimize the convex function
10: $Q_\ell^k : \mathbb{R}^{m_\ell + p - 1} \to \mathbb{R}$ defined as

$$Q_\ell^k(v_\ell, \mu_\ell) = Q(v_1^k + \mu_{\ell,1} d_1^k, \ldots, v_{\ell-1}^k + \mu_{\ell,\ell-1} d_{\ell-1}^k, v_\ell, v_{\ell+1}^k + \mu_{\ell,\ell+1} d_{\ell+1}^k, \ldots, v_p^k + \mu_{\ell,p} d_p^k)$$

11: subject to

$$(v_1^k + \mu_{\ell,1} d_1^k, \ldots, v_{\ell-1}^k + \mu_{\ell,\ell-1} d_{\ell-1}^k, v_\ell, v_{\ell+1}^k + \mu_{\ell,\ell+1} d_{\ell+1}^k \ldots, v_p^k + \mu_{\ell,p} d_p^k) \in W.$$

12: (**Synchronization**) Determine $w^{i+1} \in W$ which
13: satisfies

$$Q(w^{k+1}) \leq \min_{\ell \in \{1,\ldots,p\}} Q_\ell^k(\hat{v}_\ell^k, \hat{\mu}_\ell^k)$$

14: Set $k \leftarrow k + 1$.
15: **end while**

B. Implementation Details

1) Stopping Criterion: Applying Lemma 1 of [16], we get the following simple optimality condition:

Proposition 1: Suppose that the matrix $M(w^\star)$ is nonsingular for some $w^\star \in W$. The vector $w^\star$ constitutes a global solution to (7) if, and only if, there exists a number $\lambda^\star$ such that

$$\varphi(i, w^\star) \begin{cases} \geq \lambda^\star & \text{if } w_i^\star = 1/n, \\ = \lambda^\star & \text{if } 0 < w_i^\star < 1/n, \\ \leq \lambda^\star & \text{if } w_i^\star = 0, \end{cases} \tag{12}$$

where

$$\varphi(i, w) = \begin{cases} \operatorname{tr}\left[M^{-1}(w) S_i\right] & \text{for D-optimality,} \\ \operatorname{tr}\left[M^{-2}(w) S_i\right] & \text{for A-optimality,} \end{cases} \tag{13}$$

$i = 1, \ldots, m$.

2) Variable Distribution: As suggested in [17], the PVD-directions are chosen as the projected gradient directions, i.e.,

$$d^k := w^k - P_W[w^k - \nabla Q(w^k)], \tag{14}$$

where $P_W[w] := \arg\min_{v \in W} \|w - v\|$ signifies the orthogonal projection of w into the set W which is here the intersection of a box and a hyperplane. Due to the convexity of W, this guarantees convergence of the PVD method. Specifically, the efficient projection algorithm outlined in [18] was employed in our implementation.

3) Parallel Computations: In the k-th iteration of Algorithm 1, Processor ℓ minimizes Q_ℓ^k with respect to $v_\ell \in \mathbb{R}^{m_\ell}$ and $\mu_\ell = (\mu_{\ell,1}, \ldots, \mu_{\ell,\ell-1}, \mu_{\ell,\ell+1}, \ldots, \mu_{\ell,p}) \in \mathbb{R}^{p-1}$, subject to the constraints whose explicit form is

$$\mathbf{1}_{m_\ell}^\mathsf{T} v_\ell + (a_\ell^k)^\mathsf{T} \mu_\ell = 1 - b_\ell^k, \tag{15}$$

$$\mathbf{0}_{m_q} \leq v_q^k + \mu_{\ell,q} d_q^k \leq \frac{1}{n} \mathbf{1}_{m_q}, \quad \forall q \neq \ell, \tag{16}$$

$$\mathbf{0}_{m_\ell} \leq v_\ell \leq \frac{1}{n} \mathbf{1}_{m_\ell}, \tag{17}$$

where

$$a_\ell^k = \begin{bmatrix} \mathbf{1}_{m_1}^\mathsf{T} d_1^k \\ \vdots \\ \mathbf{1}_{m_{\ell-1}}^\mathsf{T} d_{\ell-1}^k \\ \mathbf{1}_{m_{\ell+1}}^\mathsf{T} d_{\ell+1}^k \\ \vdots \\ \mathbf{1}_{m_p}^\mathsf{T} d_p^k \end{bmatrix}, \quad b_\ell^k = \sum_{\substack{q=1 \\ q \neq \ell}}^{p} \mathbf{1}_{m_q}^\mathsf{T} v_q^k. \tag{18}$$

Here $\mathbf{0}_s$ and $\mathbf{1}_s$ denote the s-dimensional vectors of all zeros and all ones, respectively.

Optimization by Processor ℓ thus will produce the minimizer

$$\hat{w}_\ell^k = (v_1^k + \hat{\mu}_{\ell,1}^k d_1^k, \ldots, v_{\ell-1}^k + \hat{\mu}_{\ell,\ell-1}^k d_{\ell-1}^k, \hat{v}_\ell^k, v_{\ell+1}^k + \hat{\mu}_{\ell,\ell+1}^k d_{\ell+1}^k \ldots, v_p^k + \hat{\mu}_{\ell,p}^k d_p^k). \tag{19}$$

Since the objective function Q_ℓ^k is convex and the feasible set defined by (15)–(17) forms a bounded polyhedron, the local problem is perfectly suited for Simplicial Decomposition (SD), which is an important class of methods for solving

large-scale convex optimization problems, cf. [19]. It iterates by alternately solving a linear programming subproblem (the so-called *column generation problem*, CGP) which generates an extreme point of the polyhedron, and a nonlinear *restricted master problem* (RMP) which finds the maximum of the objective function over the convex hull (a simplex) of previously defined extreme points. Its principal characteristic is that the sequence of successive solutions to the master problem tends to a solution to the original problem in such a way that the objective function strictly monotonically approaches its optimal value. In our implementation, the CGP was solved using our implementation of the upper-bounding version of the revised simplex method [20, p.224], and the RMP was solved implementing an extremely simple multiplicative algorithm for optimal design in the case of the D-optimality criterion, cf. [4, p.63], and the efficient algorithm of [21] in the case of the A-optimality criterion.

4) Synchronization: Observe that, basically, for synchronization in Step 5 it would be sufficient to set w^{k+1} as $\hat{w}^k_{\ell^\star}$, where

$$\ell^\star = \arg \min_{\ell \in \{1,\ldots,p\}} Q^k_\ell(\hat{v}^k_\ell, \hat{\mu}^k_\ell) = \arg \min_{\ell \in \{1,\ldots,p\}} Q(\hat{w}^k_\ell). \tag{20}$$

Nevertheless, we decided to determine w^{k+1} as a vector minimizing the function $w \mapsto \Psi[M(w)]$ over the convex hull of the maximizers $\hat{w}^k_\ell$, $\ell = 1, \ldots, p$. Most often this leads to a better solution and the multiplicative algorithm of [4], [8, p.63] for D-optimality and the algorithm of [21] for A-optimality can be employed here again with no additional implementational efforts.

IV. COMPUTER EXAMPLE

The above idea was tested on an optimal sensor location problem for a DPS defined on the spatial domain $\Omega = [0,1]^2$. Two non-mobile sources emitting some pollutant to the air were simulated in this domain. We assume that the emission starts at $t = 0$, and the spatiotemporal evolution the pollutant concentration $y(x,t)$ is observed over the interval $T = (0,1]$. The mathematical model of the simulated process is the following PDE:

$$\begin{aligned} &\frac{\partial y(x,t)}{\partial t} + \nabla \cdot \big(\nu(x) y(x,t)\big) \\ &= \nabla \cdot \big(\kappa(x) \nabla u(x,t)\big) + f(x,t), \quad (x,t) \in \Omega \times T \end{aligned} \tag{21}$$

subject to the boundary and initial conditions

$$\begin{cases} \dfrac{\partial y(x,t)}{\partial n} = 0, & (x,t) \in \Gamma \times T, \\ y(x,0) = 0, & x \in \Omega, \end{cases} \tag{22}$$

where $\partial/\partial n$ stands for the derivative in the direction of the outward normal to Γ.

The source term

$$f(x) = a_1 e^{-100\|x-(x_{1,1},x_{2,1})\|^2} + a_2 e^{-100\|x-(x_{1,2},x_{2,2})\|^2}$$

is the sum of two terms corresponding to the two sources of pollution. They are described by two key parameters: the intensities of emissions a_1 and a_2, as well as their spatial locations $(x_{1,1}, x_{2,1})$ and $(x_{1,2}, x_{2,2})$. The coefficient $\kappa(x)$ is related to diffusion and has the form

$$\kappa(x) = b_1 + b_2 x_1^2 + b_3 x_2^2. \tag{23}$$

The vector of unknown parameters which are to be recovered based on observations is then defined as

$$\theta = (b_1, b_2, b_3, a_1, x_{1,1}, x_{2,1}, a_2, x_{1,2}, x_{2,2}). \tag{24}$$

Its preliminary estimate used to design optimal sensor locations is

$$\theta^0 = (0.05, 0.01, 0.02, 10, 0.25, 0.25, 20, 0.4, 0.7). \tag{25}$$

The wind velocity $\nu(x)$ in Ω is modelled as

$$\nu(x) = (x_2 - x_1 + 1, x_1)\,. \tag{26}$$

In order to compute the matrices S_i, $i = 1, \ldots, m$ in (3), the system equation (21) was augmented by nine sensitivity equations defining the spatiotemporal evolution of the sensitivity coefficients in (5). A procedure of forming sensitivity equations is described in detail in Ch. 2 of [4]. The resulting

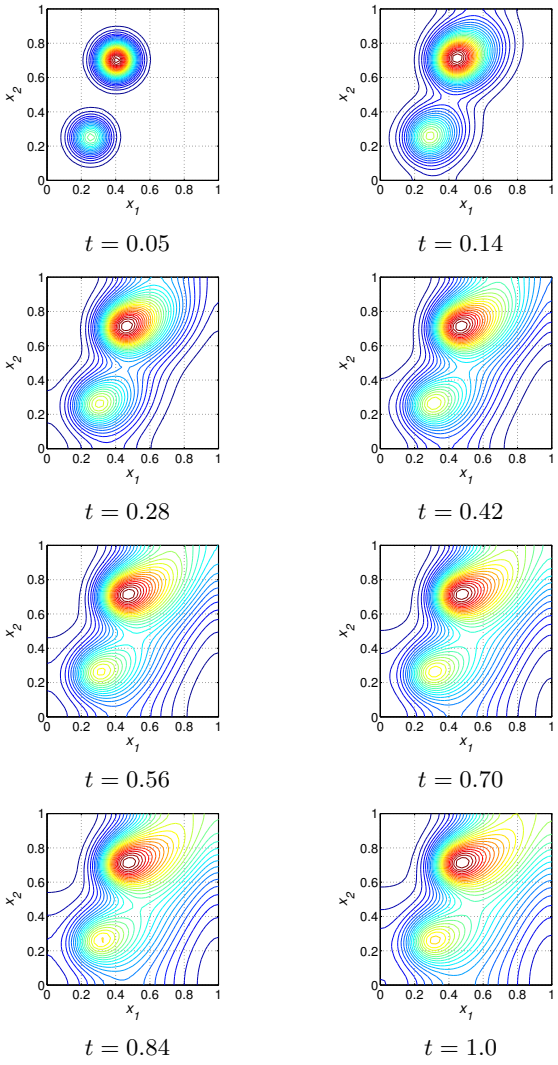
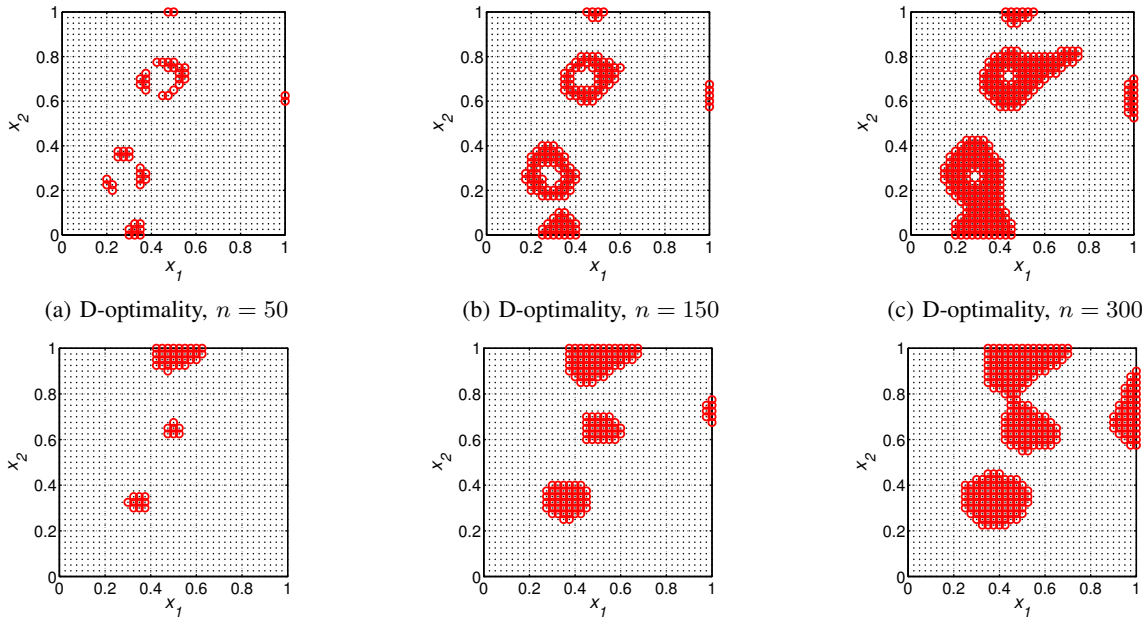


Fig. 1. Pollutant concentration at consecutive time instants.



(a) D-optimality, $n = 50$ (b) D-optimality, $n = 150$ (c) D-optimality, $n = 300$

(d) A-optimality, $n = 50$ (e) A-optimality, $n = 150$ (f) A-optimality, $n = 300$

Fig. 2. D- and A-optimum sensor configurations (top and bottom panels, respectively) for various numbers of gauged sites.

system of 10 coupled PDEs was then numerically solved using COMSOL MULTIPHYSICS v.4.3, which is based on the finite element method. Computations were performed for a grid of 3912 triangles and 2017 nodes and the time interval partitioned into 100 subintervals. The obtained sensitivity coefficients were then interpolated to get their values on the even square 41×41 grid in $[0, 1]^2$. The S_i's obtained by numerical integration were then stored as binary files and served as data for the programme determining optimal sensor configurations.

The programme implementing the PVD algorithm outlined in Section III was written in Fortran 95 and run using the Intel Fortran Compiler Professional Edition 11.1 for Linux OS, Intel Math Kernel Library and the MPICH2 implementation of the Message Passing Interface (MPI) standard. The program was run in a cluster of 32 quad-core Intel Xeon E5420 processors (altogether, up to 128 parallel nodes) on Debian Linux v.2.6.32-5-amd64.

The solution of (21) and (22) is displayed in Fig. 1. At the initial time moment, two sources of different intensities start emitting the pollutant which then spreads over the area due to diffusion and the action of the wind.

Figures 2(a)–(c) show D-optimum locations for 15, 30 and 100 sensors, respectively. For comparison, the corresponding results for A-optimum design are displayed in Figs. 2(d)–(f). As can be seen, the choice of the optimality criterion has some influence on the obtained locations. Intuitively, some part of sensors tend to measure the pollutant concentration close to the sources (but they do not cluster around the source centers). The location of the others, however, can hardly be predicted based solely on intuition, as it is influenced by a complex coupling of advection and diffusion phenomena.

The time of execution varied from several dozen seconds to several minutes, cf. Table I. Parallelization makes it shorter, which is illustrated by Fig. 3 which shows the parallel efficiency [22]

$$E_p = \frac{T_1}{pT_p}, \tag{27}$$

where T_1 is the time of execution of the best sequential solution of the problem, and T_p is the time of execution of the parallel solution of the problem on p processors, as a function of the number of processors p. Roughly speaking, E_p can be interpreted as the ratio of the actual speedup $S_p = T_1/T_p$ to the "ideal", i.e., linear, speedup [22]. Its desirable reference value is thus equal to one.

As can be observed, relatively high efficiencies were obtained for up to 30 processors. In tests for a higher number of processors, the time of execution for individual processors was rather short and this made the time of communication and synchronization rather significant compared with the time of computations.

For the example considered, the time of execution for the local subproblem is closely connected to the number of decision variables in this local subproblem. For example, when partitioning 1681 variables among 80 computing nodes, on the average each node has to operate on 100 variables, but only 21 out of these 100 variables stem from the original problem, while the other 79 variables are coordinating variables. Thus, although the speedup achieved is noticeable, we should bear in mind that the PVD algorithm presented here is primarily aimed at solving problems with large numbers of sites at which to place sensors.

V. CONCLUSIONS

A decided advantage of the PVD algorithm is its ability to operate on blocks of variables of different sizes. This can

TABLE I

TIME OF EXECUTION OF THE PVD PROCEDURE FOR DIFFERENT OPTIMALITY CRITERIA AND NUMBERS OF NODES.

Design criterion	Number of nodes / time of execution (mm:ss:ms)					
	2	8	32	48	64	80
D-optimality	05:02:796	00:49:271	00:27:435	00:19:820	00:18:289	00:17:347
A-optimality	06:15:344	00:58:537	00:29:833	00:22:972	00:19:120	00:17:820

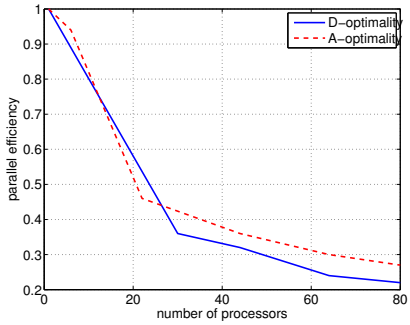


Fig. 3. Parallel efficiencies achieved.

be exploited in heterogeneous clusters of computers, where the computing power of individual nodes may substantially vary. In that case, a requirement for an efficient problem decomposition is that the work to be done by each processor be balanced among all the processors. If the work is not distributed equally, then the entire job cannot be finished until the slowest subtask is finished. But here, based on some benchmark, we can estimate the computing power of each processing node and then define blocks of variables to be processed by those nodes whose sizes are proportional to the their efficiencies of calculation. Thereby, load balancing will be guaranteed.

Even for D-optimum designs, a relatively short time of computations by no means eliminates the need for parallelization of the outlined sensor location scheme. First, the presented approach can be adopted in an on-line sensor network design. Third, the presented scheme can be adopted not only in clusters of PCs, but also on single machines equipped with multi-core processors.

Finally, with a little turning up, the presented PVD approach can be exploited to produce the solution of the relaxed problem in the branch-and-bound scheme for optimal sensor location set forth in [8]. It appears quite naturally in the B&B search for an optimal subset of gauged sites once a relaxation of 0-1 decision variables is employed to produce an upper bound to the maximum objective function. This potential embedding in the B&B scheme was one of the main motivations behind our paper and we indend to pursue this objective in our future research.

REFERENCES

[1] M. Patan, *Optimal Sensor Networks Scheduling in Identification of Distributed Parameter Systems*. Berlin: Springer-Verlag, 2012.

[2] C. Tricaud and Y. Chen, *Optimal Mobile Sensing and Actuation Policies in Cyber-physical Systems*. London: Springer-Verlag, 2012.

[3] Z. Song, Y. Chen, C. R. Sastry, and N. C. Tas, *Optimal Observation for Cyber-physical Systems: A Fisher-information-matrix-based Approach*. London: Springer-Verlag, 2009.

[4] D. Uciński, *Optimal Measurement Methods for Distributed-Parameter System Identification*. Boca Raton, FL: CRC Press, 2005.

[5] M. Patan and K. Patan, "Distributed scheduling of sensor networks for identification of spatio-temporal processes," *International Journal of Applied Mathematics and Computer Science*, vol. 22, no. 2, pp. 299–311, 2012.

[6] D. Uciński and M. Patan, "Sensor network design for the estimation of spatially distributed processes," *International Journal of Applied Mathematics and Computer Science*, vol. 20, no. 3, pp. 459–481, 2010.

[7] M. Patan and D. Uciński, "Configuring a sensor network for fault detection in distributed parameter systems," *International Journal of Applied Mathematics and Computer Science*, vol. 18, no. 4, pp. 513–524, 2008.

[8] D. Uciński and M. Patan, "D-optimal design of a monitoring network for parameter estimation of distributed systems," *Journal of Global Optimization*, vol. 39, pp. 291–322, 2007.

[9] T. Zięba and D. Uciński, "Mobile sensor routing for parameter estimation of distributed systems using the parallel tunneling method," *International Journal of Applied Mathematics and Computer Science*, vol. 18, no. 3, pp. 307–318, 2008.

[10] P. Baranowski, "Parallel and distributed processing in optimum experimental design," Ph.D. dissertation, University of Zielona Góra, Zielona Góra, Poland, 2010, (In Polish).

[11] P. Baranowski and D. Uciński, "A parallel sensor selection technique for identification of distributed parameter systems subject to correlated observations," *Lecture Notes in Computer Science*, vol. 4967, pp. 469–478, 2008.

[12] A. C. Atkinson, A. N. Donev, and R. D. Tobias, *Optimum Experimental Designs, with SAS*. Oxford: Oxford University Press, 2007.

[13] E. Rafajłowicz, "Optimum choice of moving sensor trajectories for distributed parameter system identification," *International Journal of Control*, vol. 43, no. 5, pp. 1441–1451, 1986.

[14] V. V. Fedorov and P. Hackl, *Model-Oriented Design of Experiments*, ser. Lecture Notes in Statistics. New York: Springer-Verlag, 1997.

[15] M. C. Ferris and O. L. Mangasarian, "Parallel variable distribution," *SIAM Journal on Optimization*, vol. 4, pp. 815–832, 1994.

[16] D. Uciński, "Sensor network scheduling for identification of spatially distributed processes," *International Journal of Applied Mathematics and Computer Science*, vol. 21, no. 1, pp. 25–40, 2012.

[17] C. A. Sagastizábal and M. V. Solodov, "Parallel variable distribution for constrained optimization," *Computational Optimization and Applications*, vol. 22, no. 1, pp. 111–131, 2002.

[18] N. Maculan, C. P. Santiago, E. M. Macambira, and M. H. C. Jardim, "An $O(n)$ algorithm for projecting a vector on the intersection of a hyperplane and a box in $\mathbb{R}^n$," *Journal of Optimization Theory and Applications*, vol. 117, no. 3, pp. 553–574, 2003.

[19] M. Patriksson, "Simplicial decomposition algorithms," in *Encyclopedia of Optimization*, C. A. Floudas and P. M. Pardalos, Eds. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2001, vol. 5, pp. 205–212.

[20] D. A. Pierre, *Optimization Theory with Applications*, ser. Series in Decision and Control. New York: John Wiley & Sons, 1969.

[21] N. D. Botkin and J. Stoer, "Minimization of convex functions on the convex hull of a point set," *Mathematical Methods of Operations Research*, vol. 62, no. 2, pp. 167–18, 2005.

[22] L. R. Scott, T. Clark, and B. Bagheri, *Scientific Parallel Computing*. Princeton, NJ: Princeton University Press, 2005.