Fast locational marginal pricing for congestion management in a distribution network with multiple aggregators

Citation for published version (APA):

Document status and date:
Published: 01/06/2019

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

Download date: 17. Jan. 2020
ABSTRACT

Locational Marginal Pricing (LMP) is widely used to perform congestion management in the interaction between electricity wholesale markets and transmission networks. However, the algorithm does not scale well with the number of network nodes, posing a challenge for its use for the distribution network. The problem is further complicated when multiple aggregators need to be balanced simultaneously while jointly respecting the constraints in the network. This paper proposes a fast LMP algorithm, which solves these problems. Separation of concerns between trade and grid operation is realised through an agent model, using a negotiation mechanism to reach an efficient utilisation of network capacities. Simulations with a simple network are presented to demonstrate the working principle of the algorithm.

INTRODUCTION

As the “ electrification of everything” is putting a major strain on the Western-world distribution networks, congestion management will become a standard ingredient of active distribution management. An upcoming approach is using locational marginal pricing (LMP) mechanisms for distribution congestion management [1]. Originally, LMP has been designed for the interaction between the wholesale markets and the transmission networks. Electricity distribution networks are different as their number of nodes and connected actors is much higher. The LMP algorithm does not scale well with the number of nodes in the network, which gives a problem when applied to the distribution level in the electricity system.

Another difference between the distribution and transmission networks is their topology. Where transmission networks are generally operated in a meshed topology, the distribution networks are predominantly operated in a radial, non-cyclic topology. Naturally, for reasons of security of supply, these networks are built such that each group of customers can be reached via more than one network route. However, in normal operation, the network is generally switched such that only one route is used at the time. Power flow calculations are quite straightforward in radial networks.

Making use of this property, a fast algorithm for locational marginal pricing (LMP) in radial networks has been proposed and field-demonstrated in earlier work. This algorithm, coined the Fast-LMP algorithm, was based on the idea of bid function transformations as briefly sketched in [2], has been worked out in detail in [3], chapter 7, and field demonstrated in PowerMatching City ([3], section 13.4). The algorithm combines balancing of the overall system with local congestion management. In this form, the algorithm is applicable to non-liberalized settings such as vertically integrated utility companies or islanded micro-grids operated by a single entity. In a liberalised market setting, however, functions of physical energy delivery and commodity trade are separated. Then, system operators operate the physical grids while market parties form a commodity value chain exchanging electricity. In different parts of the world, incentive mechanisms are in place to let market parties maintain real-time balance their own portfolio. In several places, market parties have begun to use distribution-level flexibility to balance their portfolios and to optimize their position in the wholesale markets. Aggregation of multiple flexible units, e.g. into virtual power plants, is a common approach. Among Distribution System Operators (DSOs) an interest is developing to use local flexibility from Aggregators to solve distribution grid congestion as well. Hence, LMP mechanisms need to be able to perform in a in a liberalized setting with multiple aggregators active in system-wide operations (balancing, market operations) while the resulting power flows meet grid capacity constraints.

In this paper, we present a novel Multi-aggregator Fast-LMP algorithm, proving a scalable adaptation of the LMP algorithm to manage congestion in electricity distribution networks with multiple aggregators. In the agent-based method, network-agnostic aggregators and portfolio-agnostic DSOs cooperate without sharing any competition-sensitive data. In the paper, we first provide formal models for the network, agents and the market. Then, we briefly describe the existing Fast-LMP algorithm, followed by the definition of the novel multi-aggregator variant (sec. 3). A proof-of-principle simulation is provided in section 4, followed by the conclusion.

NETWORK, AGENT AND MARKET MODELS

Network Model

We model a power flow network by a directed graph $G = (V, E)$, with:

- $V = \{v_1, v_2, \ldots, v_{n_v}\}$ a set of network nodes and
- $E = \{e_1, e_2, \ldots, e_{n_e}\}$ a set of directed lines with associated flow characteristics.

Each line $i$ is defined by a tuple $e_i = (h_i, t_i, r_i, z_{i, \text{max}})$, where:

- $h_i, t_i \in V$, $h_i \neq t_i$ are the head and tail nodes joined by the line. The positive flow direction is defined to
be from head $h_i$ to tail $t_i$. A negative flow value indicates a flow from tail to head.

- $r_i$ is the resistance of $e_i$,
- $z_{\text{max}}$ is the flow capacity of the line, the maximal allowable flow through $e_i$.

The lines are directed in order to define the positive flow direction: negative flow values indicate a flow against the defined line direction.

**Agent model**

There is a set of agents $X = \{x_1, x_2, ..., x_N\}$, each agent $x_k$ representing the demand and/or supply located at node $v_k$. The agent holds a demand function $d(p)$ stating the agent’s demand against resource price $p$. Each agent must act as a rational trader, i.e. its demand function $d_k(p)$ is continuous and monotonically decreasing.

**Network-agnostic market clearance**

The set of agent demand functions defines an allocation problem. As the set of agents are assumed to represent a closed system, the problem is finding an allocation of electrical power over all agents that balances demand and supply. When ignoring the network, the allocation problem is solved by finding the general equilibrium price $p^*$ such that:

$$\sum_{k=1}^{N_N} d_k(p^*) = 0 \quad (1)$$

For this price, market clearance is established, i.e. total demand equals total supply in the agent set. According to general equilibrium theory in microeconomics, the general equilibrium solution is Pareto optimal, a social optimal outcome for which no other outcome exists that makes one agent better-off without making other agents worse-off [1]. Note that, the network-agnostic market model implicitly assumes (i) all network connections are having infinite capacity (i.e. the copper plate assumption) and (ii) network losses are negligible.

**Radial Networks**

We define a radial network a network without any cycles regardless the direction of the lines. In a radial network, there is no path way starting at some node $v_k$ and following a sequence of lines for each line either in the positive or negative direction that eventually leads back to $v_k$ again. This type of network is also referred to as a tree or an acyclic network. We assume one node to be the root of the tree, with all lines directed away from the root, i.e. for each line the head is closer to the root than the tail. Accordingly, the positive flow direction is from root towards the leaves. We refer to the subtree rooted in node $k$ as subtree $k$.

**FAST-LMP: BALANCING AND NETWORK MANAGEMENT**

This algorithm achieves scalability by making only two passes through the network to come to a network feasible power flow at each network location.

**Agent Structure**

The algorithm overlays an agent & communication structure over the physical network, the agent set $X$ is organized in a tree structure following the structure of the underlying radial network. Accordingly, the agent associated with the root node of the network serves as the root agent, while each other agent has a parent agent in the direction of the root. Figure 1 shows an example network and agent structure.

**Algorithm Overview**

**Three-phase Algorithm**

The algorithm operates in three phases. In the first phase, demand functions are propagated from the leaves to the root. During this phase, demand functions are aggregated and transformed according to the local network characteristics: line capacities and transport losses. These transformations keep the demand functions network feasible for their specific location in the network. For any price value at that location, a network-feasible demand function yields a local flow that is (i) within the local line capacity constraint, and (ii) accounts for network losses. In the second phase, the root agent determines the root node price such that the market clears, i.e. total supply is equal to total demand in the network. The root agent is the only agent active in this phase. In the third phase, prices are propagated backwards to the leaves while each node agent determines the nodal price for its associated node.

In the next two subsections, we explain the bid curve transformations to account for capacity constraints and losses.

**Demand function concentration**

At each node $v_k$, agent $x_k$ concentrates its own local demand function $d_k(p)$ and, if any, the demand functions $\overline{a}_i(p)$ propagated into the node. The concentrated bid at node $v_k$ is calculated as:

$$a_k(p) = \sum_{i:e_i \in Y_k} \overline{a}_i(p) + d_k(p) \quad (2)$$

where $\overline{a}_i(p)$ is the demand function propagated to $v_k$ over line $e_i \in Y_k$. $Y_k$ is defined as:

$$Y_k = \{e_i | h_i = v_k\} \quad (3)$$

the set of lines directly connected to $v_k$ and directing away from the root of $G$. 
**Propagation of Capacity Constraints**

Assume line \( i \) has node \( k \) as tail node and \( j \) as head node and, thus, connects subtree \( k \) to the rest of the network via node \( j \). Let \( a_k(p) \) be the concentrated demand function representing all demand and supply in the subtree. Then, all prices \( p \) for which \( |a_k(p)| > z_{i,\text{max}} \) will overload line \( i \). To prevent this, \( |a_k(p)| \) is limited to \( z_{i,\text{max}} \) for exactly these prices. An example of this alternation is shown in Figure 2. More formally, the propagated demand function \( \overline{a}_k(p) \) for a line capacity constraint is calculated as:

\[
\overline{a}_k(p) = \begin{cases} 
  z_{i,\text{max}}, & z_{i,\text{max}} < a_k(p) \\
  a_k(p), & -z_{i,\text{max}} \leq a_k(p) \leq z_{i,\text{max}} \\
  -z_{i,\text{max}}, & a_k(p) < -z_{i,\text{max}} 
\end{cases}
\]  

(4)

The resulting demand function is network feasible for line \( i \) and is propagated over this line. In this way, only the network-feasible contribution of subtree \( k \) to the system-level demand/supply balance is taken into account in the price forming at the root node. A similar transformation can be defined to incorporate line losses.

**Root Nodal Price**

The price at the root node \( (v_{N_0}) \) is chosen such that the market at the root node is in equilibrium:

\[
a_{N_0}(p_{N_0}) = 0 
\]  

(5)

where \( a_{N_0}(p) \) is the concentrated demand function for the root node.

**Price Back Propagation**

The price at the root node is then propagated back along each line in the tree network. At each node \( v_k \), the nodal price \( p_k \) is determined from the nodal price at \( j \) as shown in Figure 3. Input to this process are the original, untransformed, demand function \( a_k \) and price \( p_j \) at parent node \( j \). The figure shows three distinct cases:

(A) Price \( p_j \) does not cause line overloading. Price \( p_k \) is set to \( p_j \).

(B) Price \( p_j \) will result in an overloading demand level. Price \( p_k \) is set to the price that results in a demand equal to the line capacity \( z_{i,\text{max}} \).

(C) Similar to (B) for an overloading supply level.

Note that, in all cases and for all nodal prices \( p_j \), the following equality holds:

\[
a_k(p_k) = \overline{a}_k(p_j) \]  

(6)

where \( \overline{a}_k(p) \) denotes the propagation of demand function \( a_k \). In other words, the resulting demand at \( k \) is equal to the propagated demand function at the nodal price at node \( j \). More formally, \( p_k \) is calculated as:

\[
p_k = \begin{cases} 
  p_j, & |a_k(p_j)| \leq z_{i,\text{max}} \\
  p_k; a_k(p_k) = z_{i,\text{max}}, & a_k(p_j) > z_{i,\text{max}} \\
  p_k; a_k(p_k) = -z_{i,\text{max}}, & a_k(p_j) < -z_{i,\text{max}} 
\end{cases} 
\]  

(7)

**MULTI-AGGREGATOR NETWORK MANAGEMENT**

In this section, we enhance the algorithm to let it operate in a multi-aggregator setting. To do this, we (i) split off the network management function into a separate grid agent, and (ii) split the balancing function into a number of market players (i.e. aggregators) each running an aggregation architecture of their own.

**Revisited Agent Model**

There is a set of aggregators \( A = \{a_1, a_2, ..., a_{N_a}\} \) and each aggregator \( a \) has an associated set of **portfolio agents** \( X = \{x_{a,1}, x_{a,2}, ..., x_{a,N_a}\} \). Each agent \( x_{a,k} \) represents the aggregator’s **portfolio** at location \( v_k \), i.e. all demand and supply as associated with aggregator \( a \) in the network subtree rooted in node \( v_k \). Each consumer, producer or prosumer is connected to exactly one aggregator, as determined by their energy contract. Further, there is a set of grid agents \( V = \{v_1, v_2, ..., v_{N_v}\} \), each agent \( v_k \) representing the grid at network node \( v_k \).

Figure 4 shows an example agent topology for a four-node network with two aggregators. Each network node \( v_v \) has a grid agent associated with the same identifier. Each aggregator \( a \) is represented by a tree-shaped topology of local portfolio agents \( x_{a,v} \), where agent \( x_{a,v} \) represents the portfolio of aggregator \( a \) at location \( v \).

**Algorithm**

**Objective**

Reach an pareto optimal overall demand and supply balance for each of the aggregator portfolios, collectively meeting with network constraints, while optimally using network capacity.

**Algorithm Phases**

Similar to the single aggregator case, this algorithm has
three distinct phases: (1) upward propagation of bids, (2) price forming for portfolio balancing, and (3) price back propagation. There are two main differences from the single aggregator case. Firstly, the price forming in the second phase happens for each aggregator individually. The resulting price is dependent on the specific portfolio of the aggregator and, thus, each aggregator’s root price may be different. Secondly, in the multi-aggregator case, the algorithm may have an additional phase to determine optimal utilisation of network capacity.

**Phase 1: Upward propagation of bids**

At the start of this phase, the local portfolio agents $x_{a,v}$ collect all their incoming demand functions and aggregate these to demand function $a_{a,v}$. All resulting functions are then sent to the grid agent $v$, who aggregates them into one, resulting in a full demand function of all demand and supply in the $v$ sub-tree. The grid agent then applies the transformations as in the single aggregator case to this function. It then communicates back to the portfolio agents the congestion boundary prices $p_{v,min}$ and $p_{v,max}$ that contain the network-feasible flow $z_{i,max}$ determined by:

$$p_{v,min} = \max_{a} a_{v}(p), \text{ subject to: } a_{v}(p) \leq z_{i,max}$$

$$p_{v,max} = \min_{a} a_{v}(p), \text{ subject to: } a_{v}(p) \geq -z_{i,max}$$

The portfolio agents then transform their respective aggregated functions $a_{a,v}$ to $a'_{a,v}$ as follows before propagating it upwards:

$$a'_{a,v}(p) = \begin{cases} a_{a,v}(p), & p \leq p_{v,min} \\ a_{a,v}(p), & p_{v,min} < p < p_{v,max} \\ a_{a,v}(p), & p \geq p_{v,max} \end{cases}$$

where, $a_{a,v}(p_{v,min}) = z_{a,v,max}$ and $a_{a,v}(p_{v,max}) = z_{a,v,min}$ represent the allotted capacity for aggregator $a$ at node $v$.

**Phase 2: Price forming for portfolio balancing**

All root portfolio agents $x_{a,1}$ collect their incoming demand functions, aggregate these, search for equilibrium price $p_{a}^*$, and passes this price back to their own child agents. For each aggregator $a$, price $p_{a}^*$ balances its entire portfolio. Note that each price can differ from the others as the price depends on the aggregator’s portfolio of devices.

**Phase 3: Price back propagation**

All portfolio agents $x_{a,j}$ receive a price $p_{a,j}$ from their parent agent at node $j$. There are three possibilities:

Case 1: All portfolio agents have their price in the uncongested area of their bid curve. Then the grid agent signals an OK to the portfolio agents who then pass their price on to their children.

Case 2: All portfolio agents have their prices in the congested price area on one side of the price range (i.e. either all high or all low prices). Then, the portfolio agent alters each aggregator price $p_{a,j}$ into the price $p_{a,v}$ using its transformed function $a_{a,v}$ according to (8). These altered prices are passed to their children.

Case 3: Some of the portfolio agents do use their own grid cap, while others don’t. In this case, unused grid capacity can be shifted from the unconstrained aggregators to the ones that are constrained.

**Phase 4: Capacity negotiation iteration**

In the event of case 3, unused capacity $z_{v,+}$ at network node $v$, is assimilated by the grid node agent from all aggregators who have local prices in their unconstrained bid region ($a \in A^+ \subseteq A$ for which $p_{a,v} \in [p_{v,max}^i, p_{v,min}^i]$) as:

$$z_{v,+} = \sum_{a \in A^+} z_{a,v,max} - a_{a,v}(p_{a,v}), a_{a,v}(p_{a,v}) \geq 0$$

Similarly, desired capacity $z_{v,-}$ is also calculated by the grid agent from all aggregators who have local prices in their constrained bid region ($a \in A^- \subseteq A$ for which $p_{a,v} \notin [p_{v,max}^i, p_{v,min}^i]$) as:

$$z_{v,-} = \sum_{a \in A^-} a_{a,v}(p_{a,v}) - a_{a,v}(p_{a,v})$$

Then, the following iteration is performed:

Step 1: The excess capacity, $\min(z_{v,+}, z_{v,-})$ is equally divided to all aggregators $a \in A^-$. 

Step 2: For all aggregators $a \in A^-$, redo phases 1 through 4.

Step 3: If $z_{v,+} > 0$ and $z_{v,-} > 0$, go to Step 1.

**SIMULATION EXAMPLE**

The Fast-LMP algorithm driving the portfolio balancing (for a single round) in a simple electrical grid consisting of two industrial energy producers and four domestic consumers was simulated in MATLAB. The producers/consumers are assumed to be affiliated to two aggregator services each, as shown in Figure 5. Also present in the grid are two transformers, one in each of the consumer branches that are representative of capacity bottlenecks. For this simulation, the bid curves of all portfolio agents are chosen to be fictitious but representative of demand functions of the producer/consumer they represent. Initially, allotted network capacity in nodes $v_2$ and $v_3$ are as shown in Figure 6 (Left) and Figure 7 (Left). It can be seen from Figure 6 (Right) that the local prices of Aggregator 1 agents $x_{1,2}$ and $x_{1,3}$ have resulted in allocations which do not use its allotted network capacity entirely. Whereas in
Aggregator 2, the local prices of agents $x_{2,2}$ and $x_{2,3}$ result in allocations which use all of their allotted network capacities.

**Figure 5:** Hypothetical network with portfolio agent $x_{mn}$ where $m$ is the aggregator they are affiliated to, and $n$ is the branch in the topology where they belong.

Furthermore, these agents could benefit from some extra capacity to achieve a more optimal balance according to the original demand functions. Thus, the unused capacity from Aggregator 1 is transferred to Aggregator 2 resulting in a set of enhanced network constraints (as seen in Figure 8 (Left)) for the latter. Subsequently, Aggregator 2 performs a second round of portfolio balancing, resulting in a balance (Figure 8 (Right)) well within the network constraints, in accordance with the original demand functions.

**Round 1:**

**Aggregator 1**

**Aggregator 2**

**Round 2:**

**Aggregator 1**

All agents from Aggregator 1 are already in balance.

**Aggregator 2**

**Figure 6:** (Left) Bid functions of portfolio agents of Aggregator 1 without the effect of network constraints at $V_{2,2}$ and $V_{2,3}$. (Right) Bid functions after considering constraints. Root price is within uncongested region of all bid functions. This results in unused capacity in nodes $V_{2,2}$ and $V_{2,3}$.

**Figure 7:** (Left) Bid functions of portfolio agents of Aggregator 2 without the effect of network constraints at $V_{2,2}$ and $V_{2,3}$. (Right) Bid functions after considering constraints. Local prices for agents $x_{2,2}$ and $x_{2,3}$ were shifted from the congested region of their respective bid functions. They can benefit from some extra capacity.

**Figure 8:** (Left) Bid functions of portfolio agents of Aggregator 2 with enhanced capacity constraints thanks to capacity transferred from Aggregator 1. (Right) Prices are now in uncongested region. Total flow in either branch is within constraints and network capacity is better utilised.

**CONCLUSION**

Fast-LMP proves to be a scalable adaptation of the LMP algorithm to manage congestion in distributed electricity networks with single and multiple aggregators. With the agent-based approach, it is possible for network-agnostic aggregators and portfolio-agnostic DSOs to cooperate in solving problems in the distribution network without sharing any competition-sensitive data. The capacity negotiation phase ensures that the network capacity is efficiently utilised and consumers would not have to compromise comfort in the name of congestion mitigation. The simulation shows promise as a first proof of concept of the algorithm and paves the way for developing an agent-based communication framework to be tested with real world scenarios.

**REFERENCES**


**ACKNOWLEDGMENTS**

The work presented in this paper was partially financed by the TNO Early Research Program on Complexity (TEBAT Project) and the OPZuid project “Smart Energy Management”.