Distributed Optimization and Machine Learning

Prof. Mayank Baranwal

Computer Science & Engineering, Electrical Engineering, Mathematics

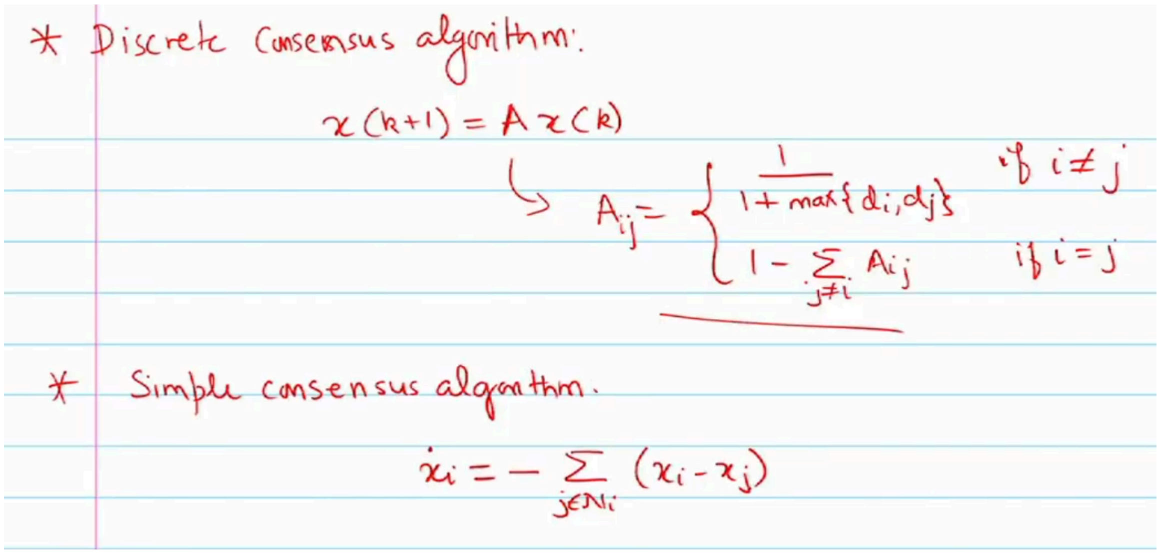
Indian Institute of Technology Bombay

Week-9

Lecture 32: Distributed Economic Dispatch Problem

In the last lecture, we looked at different consensus algorithm. So, we had the discrete version of it. So, let us discrete consensus algorithm. which was of the form xk plus 1 is a times xk and this a happens to be let us say using the metropolis Hastings weighting algorithm. So, a was designed to be of the form max of d i. So, this is a ij.

and this is only when ij, i and j are neighbors. This is how you design your consensus algorithm right for discrete time. Then we looked at the continuous time variance of it. So, there was a simple a standard consensus algorithm which was every agent i would run this x i dot is x i dot is essentially summation x i minus x j, but the summation is going to be over the neighborhood set of i right.



* Discrete Consensus algorithm:

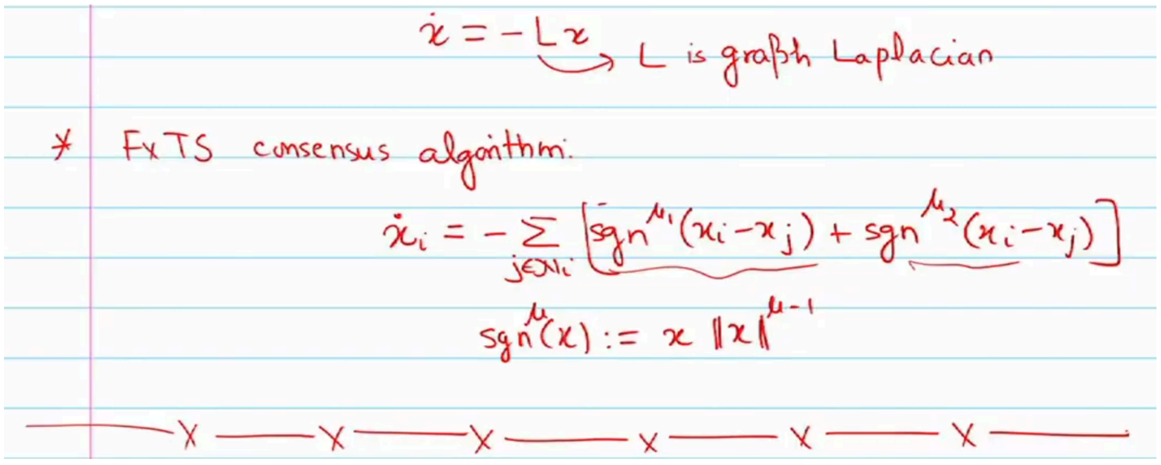
$$x(k+1) = A\,x(k)$$

$$\hookrightarrow \quad A_{ij} = \begin{cases} \dfrac{1}{1+\max\{d_i, d_j\}} & \text{if } i \neq j \\ 1 - \sum\limits_{j\neq i} A_{ij} & \text{if } i = j \end{cases}$$

* Simple consensus algorithm.

$$\dot{x}_i = -\sum_{j\in N_i} (x_i - x_j)$$

So, if I were to write this vectorized form of it, what would this look like? x dot is negative L of x right, where L is the Laplacian And finally, we looked at the fixed time variant of it. So, it was and the idea behind fixed time is. So, you we use x i dot is negative this time we use this funny looking function of this form right. So, instead of using xi minus xj we had this kind of consensus scheme right and how do we define sin of x is defined to be is how you define this right and we choose mu 1 and mu 2 to be of basically mu 1 is a fraction and mu 2 is a fraction. So, these are the numerator and denominator are the odd terms right.



$$\dot{x} = -Lx \xrightarrow{} L \text{ is graph Laplacian}$$

* FxTS consensus algorithm:

$$\dot{x}_i = -\sum_{j\in\mathcal{N}_i}\left[\underbrace{sgn^{\mu_1}(x_i-x_j)} + sgn^{\mu_2}\underbrace{(x_i-x_j)}\right]$$

$$sgn^{\mu}(x) := x\,\|x\|^{\mu-1}$$

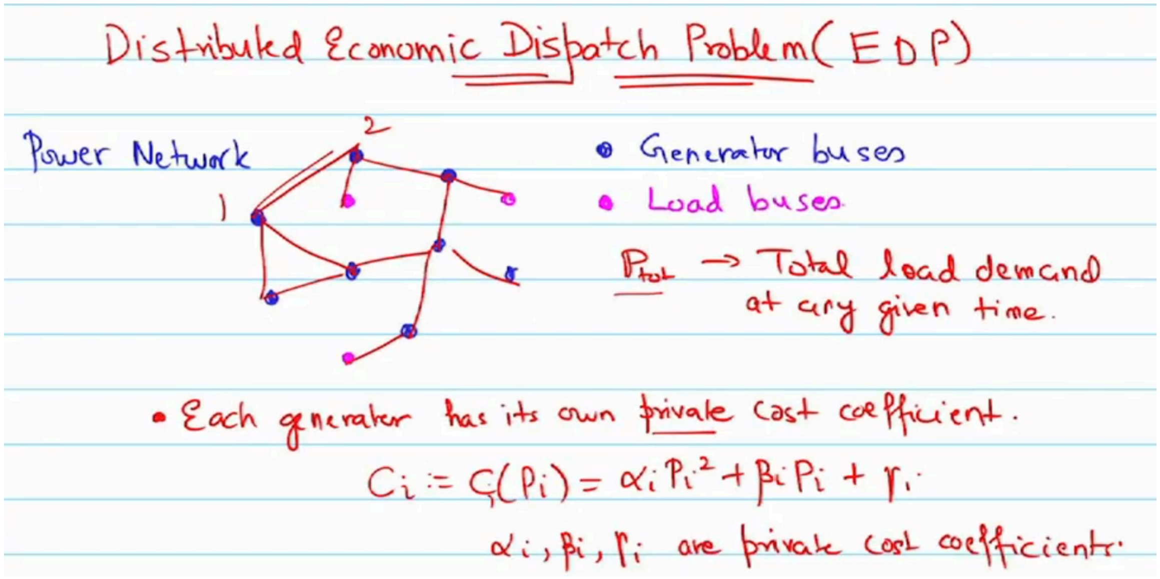
So, you basically make these functions as odd functions and if you try to recover pretty much something like this which is the case with standard consensus algorithm, but then you because we want to expedite it even further that is where we use this particular kind of construct. What we have seen so far, today would be the first application of this consensus scheme and in fact your first introduction to real world distributed optimization problem which is distributed economic dispatch problem. I will write this as EDP for economic dispatch column. And the idea is, so does anyone recall what is what is economic dispatch from previous lectures or the first lecture rather the introductory lecture we had. So, this is about a power network and you have multiple types of buses in the network.

Let me use a different color and so on and what may happen is, so you have a network which is connected like this and so on. So, you think of these blue dots as generator versus this is an power network. So you have generator buses, you think of these sign dots or the magenta dots as load buses and the objective is, so essentially let us say at any given time, so at any given time you have p total as the total load demand at any given time. ok. So, every generator it would want to generate certain power.

So, that the total power that is generated in the network, it is going to be equal to this p total, this quantity that we want to match right. You do not want to produce extra because in that case you would be I mean it would either go as line losses or you would end up damaging certain parts of the network obviously or may be in fact the loaded loads that are going to be that are connected to the network. Or you are also not going to be producing lesser than what is demanded because in that case you would not be able to serve the load total load demand that is there right. Now, when I talk about this kind of underlying network in this example, you see these are the edges right. So, they can be I mean you can basically manifest the physical connection between two generators as these edges or you can also assume that a generator let us say I call the generator 1 and generator 2.

So, that means generators 1 and 2 can exchange information with each other right. Now, each generator, so another constraint is that each generator has its own private cost coefficient. Now suppose you are a independent system operator who runs this power network right and you ask different generators to generate power so that you meet the total load demand. Now from the perspective of the generator, I would want to generate as much power as I mean as I can, so that all the profits that I am going to be getting like after meeting the demand, I mean that should get credited to me right. So, I would ideally want to not care about what everyone else is doing, but I would I mean ideally like to minimize my own or maximize my own profit right.

And that is against the sort of central theme of this distributed cooperative optimization that we want to focus on. So, we want to be able to generate like every generator want to be able to generate the required amount of power in such a manner so that the sum total is going to be equal to P total this quantity P total. But then at the same time, I am not like aware of what other generators private cost coefficients are right. So, essentially, if the cost of generation, let us call it Ci or defined as C of Pi, Ci of Pi, this is going to be alpha i Pi square. So, we assume it is usually it is assumed it is a quadratic cost.


Distributed Economic Dispatch Problem (EDP)

Power Network

● Generator buses

● Load buses

P_{tot} → Total load demand at any given time.

• Each generator has its own private cost coefficient.

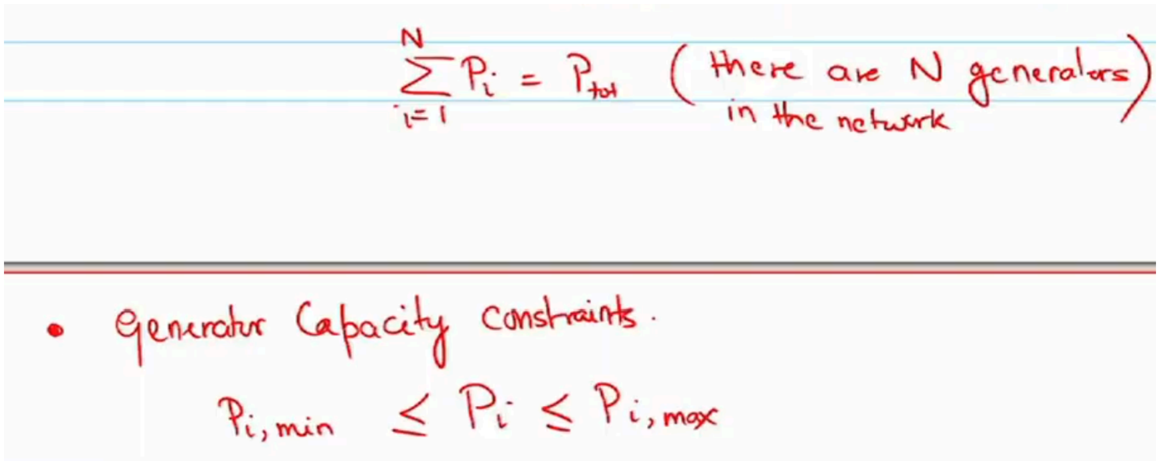
$$C_i := C_i(P_i) = \alpha_i P_i^2 + \beta_i P_i + \gamma_i$$

$\alpha_i, \beta_i, \gamma_i$ are private cost coefficients.

So, these coefficients like for the ith generator these coefficients alpha i beta i and gamma i they are only known to the ith generator and they are not known to the other generators. So, that is why it is private cost coefficient. So, I would not want to tell my neighboring generator about how much it takes for me to generate a certain amount of power right, because anyone else can then sort of use that information and try to maximize their gains. So, you would ideally want to keep your private cost like basically cost coefficients private. So, only you know how much it costs you to generate certain amount of power.

So, that is one constraint that the total power generated is should be equal to p total. So, essentially there are n generators in the network. So, this is one constraint right pi this is going to be equal to p total. So, there are n generators in the network ok. what else can you think of in terms of constraints for this problem? No, that is the objective, but in terms of constraints what else can you think of? So, for each generator for instance has certain like I mean it cannot generate beyond its capacity right.

So, there are going to be some capacity kind of constraints. So, there are going to be generation a generator capacity constraints. So, let us say the higher generator cannot generate anything more than pi max and it has to produce at least pi min amount. So, in certain cases we can assume that pi min is 0. So, either it produces nothing or it produces not more than something like pi max.



$$\sum_{i=1}^{N} P_i = P_{tot} \quad (\text{there are } N \text{ generators in the network})$$

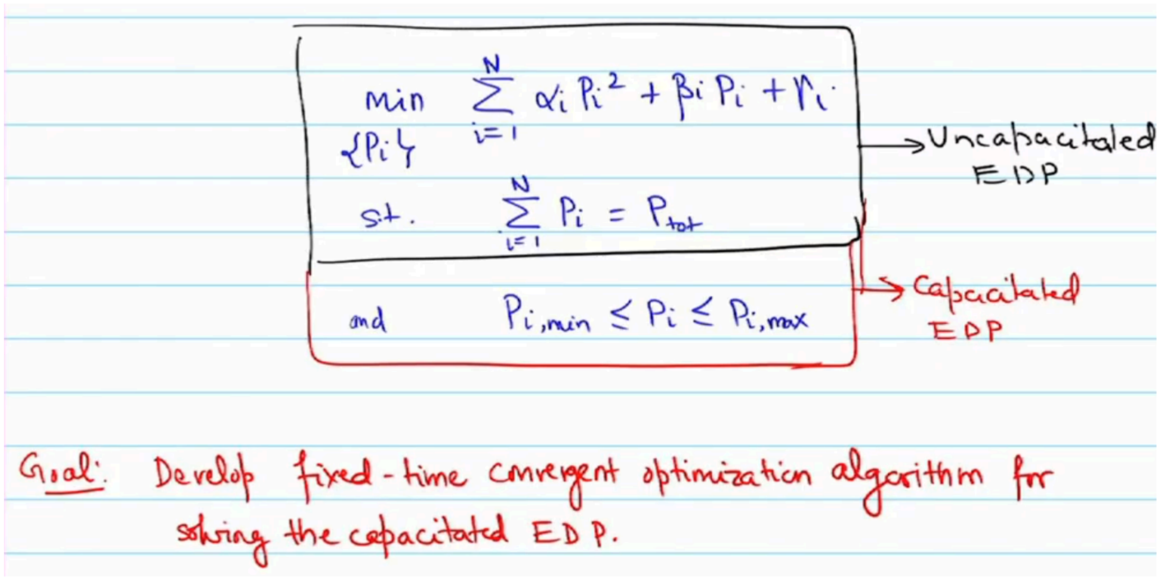
- Generator Capacity constraints.

$$P_{i,min} \leq P_i \leq P_{i,max}$$

So, this is the capacity constraint. So, what should be the social goal here or what should be the cumulative goal here? So, to minimize the total cost generation right, I do not care about a particular generator, I want to minimize the total cost of generation subject to certain constraints, one is the demand constraint. So, that total generation should be equal to the total demand that is one constraint and the other constraint is the capacity constraints right. So, this basically allows us to write down the economic dispatch problem in and this is. with respect to pi summation 1 through n ok.

So, this is your objective function subject to you have 1 through n pi this is equal to the total load demand. you have the capacity constraint. So, we so far in this course we have mostly focused on unconstrained optimization as far as minimizing a particular objective function is concerned. I mean we had also looked at equality constraint optimization in the context of Lagrangian based methods and so on, but so far we have largely focused on unconstrained optimization. So, if I just look at could care about this particular problem and ignore this capacity constraint ok.

So, this is called uncapacitated economic dispatch problem. I mean it still has equality constraint, but then equality constraints we know are easier to work with than the inequality constraint. So, this is usually called uncapacitated economic dispatch problem. And the moment we also include the capacity constraints, then this is called the capacitated the entire problem, this is called the capacitated economic dispatch problem. And the goal is at least for this lecture the goal is so develop fixed time convergent optimization algorithm.



$$\min_{\{P_i\}} \sum_{i=1}^{N} \alpha_i P_i^2 + \beta_i P_i + \gamma_i$$

$$\text{s.t.} \quad \sum_{i=1}^{N} P_i = P_{tot}$$

$\rightarrow$ Uncapacitated EDP

$$\text{and} \quad P_{i,min} \le P_i \le P_{i,max}$$

$\rightarrow$ Capacitated EDP

Goal: Develop fixed-time convergent optimization algorithm for solving the capacitated EDP.

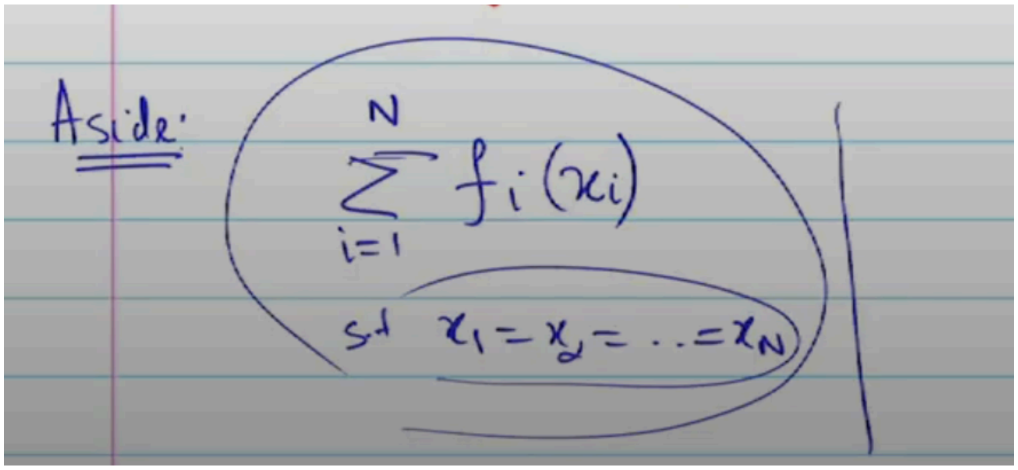
for solving the capacitated economic the full version or the capacitated economic dispatch I mean the agents know, the problem is well posed, I mean the problem is well defined, you do not know as a centralized entity, you do not know, I mean we have not come to the algorithm part yet right, the problem is well posed. I mean the algorithm what how we are going to be ensuring that this gets satisfied in the sense that only agent i needs to know about alpha beta and gamma, but they would they are still able to solve this problem. So, that is so obviously they are going to exchange some amount some information with their neighbors right. And, that is like if let us say in a distributed optimization problem, any node let us say there are multiple nodes in the network. And, if there is no exchange of information either in the consensus problem or the optimization problem, there is no exchange of information between neighbors, there is no way you can reach the centralized goal right.

One way, one thing is to not know about what someone else is trying to minimize, but you can also, you can extend certain other information. Let us say you can, let us say you are trying to minimize summation of fi, fi xi right. I need not know what their fi is, but I can at maybe query what your current xi is and based on that I can try to get a sense of what the cumulative objective function looks like and that is what we would be focusing on. So, in this case we assume that we do not know alpha, like like agent j does not know what or the generator j does not know alpha, beta and gamma for the i h generator, but they would still want to minimize this total objective function, which is the sum of the individual generation cost subject to the equality constraint. And if you include the capacitor economic dispatch problem, then it also I mean you also in basically include the end up including the generation capacity constraints right.

So, as I said inequality constraints are relatively difficult to deal with. So, to I mean to make things easier we would first start with the uncapacitated economic dispatch problem. We will try and develop an algorithm for that where we would guarantee that every agent would converge to the optimal solution for this. So, this is slightly different from the usual economic, usual distributed optimization problem right.

Suppose I want to minimize. So, this is just an aside thing. So, some of the variance that we have looked at for the economic or for the usual distributed optimization problem is subject to x1 equal to x2 equal to x3 and xn right. That is what we had looked at right so far at least that is how we had introduced distributed optimization problem. So, this is this particular problem is slightly different from the problem over here because we do not want this consensus kind of constraint on x i's. because I mean every generator can end up generating different quantity different amount of power right.

We just want to ensure that the whatever power that they are generating the sum total of it is going to be equal to the total load demand. So, it is slightly different from that, but again like it needs to be solved in a distributed manner and because it is a distributed optimization problem. So, it sort of falls under the same similar category alright. So, how do you think are the agents or the generators in this case are going to be coupled with each other? are they coupled in this column? Let us say, so let us focus on uncapacitated economic dispatch volume for now. So, we are going to look at uncapacitated economic dispatch volume.



And looking at this formulation, how do you think the different generators are coupled with each other? If I just look at the uncapacitated column, So they are not coupled through objective function. They are coupled through the constraint which is like if I mean you cannot greedily generate or greedily optimize for your own objective function because we also have to ensure this particular thing. If had there been no constraint then everyone would try to optimize their own objective function locally and that this would have been fine. But because there is this constraint here that means I cannot just look at I mean greedily sort of optimize my own objective function and not care about someone else's So, if you have an equality constraint like this and you do not want to exchange your cost coefficients with your neighbors because those are your private cost coefficients. What variable or what can you think of as a proxy or something that you can exchange with your neighbors to get meaning to get the to get to the relevant or to be able to minimize the objective function here.

When we think of this constraint. p total minus pi, it would not help as much. So, when you have equality constraint problems, how do you deal like convert them into unconstrained optimization? Use the Lagrangian formulation. So, use the Lagrangian. So, what is the Lagrangian of this problem? So, let us say there is a Lagrange multiplier lambda, it is going to be a scalar variable right, because there is just one equality constraint. So, it is going to be a scalar variable.

So, it is going to be defined in terms of pi's and lambda and this is going to be your original objective function. So, this is your Lagrange. And we are defining the Lagrangian because it is equality constrained optimization problem. So, we are trying to convert this constrained optimization problem into an unconstrained one right plus ok. If you want to use augmented Lagrangian then you also add square of it or something, it is up to you how you want to.

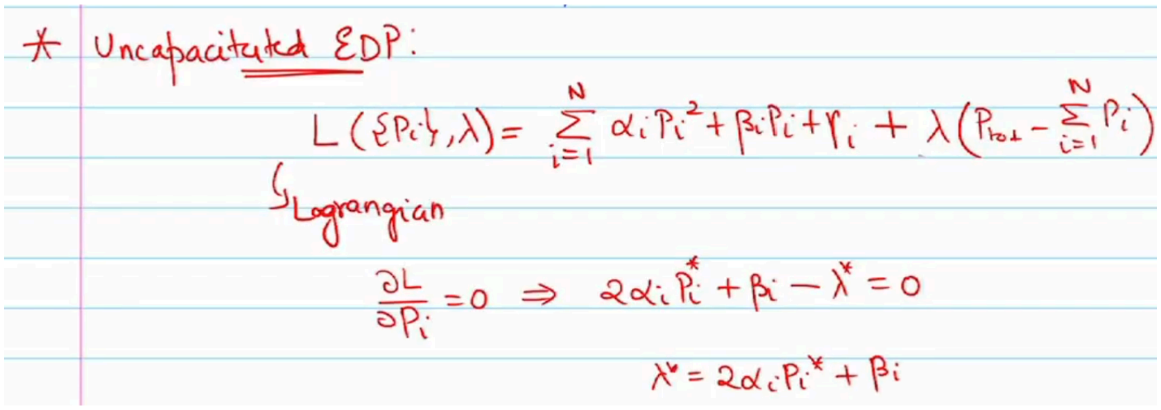
But what is one special thing about this formulation other than like I mean you have converted the unconstrained the constraint problem into an unconstrained problem. What is one special thing about this formulation? What is one special thing about the dual variable or the Lagrange multiplier lambda here? Let me also I mean just to be consistent with sign I mean it does not matter. So, I am going to be writing this is I mean this is just to be consistent with the standard text ok. I mean it is one and the same thing because it is both I mean these are equality constraints.

So, it is one and the same thing. No, but p total is a demand right, p total is not the total of what everyone, even if it is known, I am saying that p, well let us assume that it is not known, but let us say even if it is known, you also have to ensure that you minimize this plus, I mean sum, so p total is a demand, it is not what everyone else is generating. Like it is not the sum total of what everyone is generating. This is the demand, load demand right, so you want to, it may vary, but I am saying that like do not think of p total as summation of p i right. I mean it may not be right, if you start with you will be very different from it.

Yes sir, they do not care about p total. But what is lambda? So, one thing is that when you look at this dual variable right. So, what kind of information are you going to be exchanging with your neighbor and it is much easier to exchange this lambda right. So, I would say that I have my own belief on lambda you my neighbor may have. So, essentially it is going to we are going to be running consensus on lambda. And so, somehow we have to use a distributed optimization or distributed consensus flavor to be able to I mean we have to reach a common goal right.

And of the things that we know, we are essentially going to be running consensus on lambda because everyone shares the same lambda. Is this clear? So, let us look at, let us try to characterize this lambda. Yes, I mean, for now, let us assume it is no, I mean, but then eventually, I would say eventually no, but let us try and characterize this lambda first. So, how do we find lambda? So, this is an unconstrained optimization polymer alright. So, the derivative of this with respect to pi as well as lambda should be 0.

So, this when you take the derivative with respect to lambda set it to 0, this is you recover nothing but the equality constraint. But if I take the derivative with respect to pi and set it equal to 0, this gives me 2 alpha i pi plus beta i minus lambda, let us say this is at optimality right. lambda star this should be equal to 0 or this gives me lambda star is equal to 2 alpha i pi star plus beta i. In other words pi star first, so let us first let us call it equation 1 and if I look at pi star from here. this is nothing but lambda star minus beta i upon 2 alpha i.



* Uncapacitated EDP:

$$L(\{P_i\},\lambda) = \sum_{i=1}^{N} \alpha_i P_i^2 + \beta_i P_i + \gamma_i + \lambda\left(P_{tot} - \sum_{i=1}^{N} P_i\right)$$

↳ Lagrangian

$$\frac{\partial L}{\partial P_i} = 0 \Rightarrow 2\alpha_i P_i^* + \beta_i - \lambda^* = 0$$

$$\lambda^* = 2\alpha_i P_i^* + \beta_i$$

So, if I can reach consensus on lambda, I can just subtract beta i from there divided by 2 alpha i and that would be the optimal power that I should be generating. So, this is one thing. So, this basically tells you that the generators they would actually want to run consensus on lambda. Once they arrive at a consensus on lambda, they would ensure they would just ensure that their p i star is essentially going to be this.

So, they do not need to worry about. Now, what is lambda? Let us say what is lambda star? So, how do you find lambda star from here? So, if I divide lambda star by 2 alpha i, this is p i star plus beta i over 2 alpha and I know that summation p i star is going to be p total. So, that means lambda star times summation i equal 1 through n 1 over 2 alpha i. So, this is equal to p total plus summation i equal 1 through n beta i over 2 alpha i. this implies lambda star for uncapacitated economic dispatch problem. So, this is going to be p total plus summation i 1 through n.

$$P_i^* = \frac{\lambda^* - \beta_i}{2\alpha_i} \quad \text{—— ②}$$

$$\frac{\lambda^*}{2\alpha_i} = P_i^* + \frac{\beta_i}{2\alpha_i}$$

$$\lambda^* \left(\sum_{i=1}^{N} \frac{1}{2\alpha_i} \right) = P_{tot} + \sum_{i=1}^{N} \frac{\beta_i}{2\alpha_i}$$

$$\boxed{\lambda^*_{un} = \frac{P_{tot} + \sum_{i=1}^{N} \frac{\beta_i}{2\alpha_i}}{\sum_{i=1}^{N} \frac{1}{2\alpha_i}}}$$

So, this is how you characterize if you were to solve this in a centralized fashion. this would have been your lambda star. But I mean because we are not going to solve it in a centralized fashion, we are just going to run consensus on lambda star and then eventually use this to find pi star. Then how do we ensure that this equality constraint is satisfied? So you are going to be generating initial pi 0 in such a manner so that sum of pi 0 is going to be p dot. So, as long as that is satisfied, because the sum is preserved, you are going to get lambda star and once you get the lambda star, then you are just going to sum it up.

you are just going to run equation 2 and that is how you are going to get pi star. So, now in exchanging this information about lambda star, I do not have to inform my neighbors about my private cost coefficients right. All I am going to be exchanging information with my neighbors is just lambda, the dual variable. And once I know that there is a consensus on lambda, even if there is no consensus on lambda, every time I am going to be generating pi, which is going to be lambda minus beta over 2 alpha right. And as soon as this converges that is that means we have reached the consensus and this is the going to be the optimal dispatch value from the generator.

Is this clear? Yeah. So, let us say for instance everyone like there are n generators. So, for now assume that p total is known. So, if there are n generators everyone is going to be generating p total p total by n. That is one way.

The other way is and which is the right way. So, the load is going to be connected to one of the generator buses. So, the load bus is going to be. So, essentially the sum of these load values is going to be p total. So, this load bus is simply going to relate to its nearest generator about its total load demand. And this would be the p i 0 for this, this would be the p i 0 for this, this would be the p i 0 for this.

Because no other load is communicating its demand to any of these generators, their p i's p 0's would be 0. So, in that case summation p i 0 is going to be guaranteed to be equal to p total. So, that is So, you do not need to know the entire load in the network, the generate the load buses are just going to communicate their to their nearest generator buses about the total load demand. It may happen that there are certain scenarios for instance like you have multiple loads connected to load buses connected to the same generator. So, in that case this generator pi 0 is going to be the sum of these two ok.

But then in this way you know the total load demand in the network. you do not I mean you do not know the total load demand in the network, but you are just going to be ensuring that summation pi 0 is always going to be p total. So that is one thing we had already seen right in when you run the average consensus it is its average consensus right. So summation pi is always going to be equal to summation pi 0 that quantity remains conserved.

So we have to design algorithm such that this is going to be true. So we are now going to look at the algorithm. Thank you very much.