ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΗΛΕΚΤΡΙΚΗΣ ΙΣΧΥΟΣ

# Εφαρμογή της τεχνολογίας Blockchain για την επίλυση του προβλήματος Οικονομικής Κατανομής Φορτίου

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ηλίας Κάππος

**Επιβλέπων : Νικόλαος Χατζηαργυρίου**

Καθηγητής Ε.Μ.Π.

Αθήνα,Οκτώβριος,2019

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΗΛΕΚΤΡΙΚΗΣ ΙΣΧΥΟΣ

# Εφαρμογή της τεχνολογίας Blockchain για την επίλυση του προβλήματος Οικονομικής Κατανομής Φορτίου

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ηλίας Κάππος

**Επιβλέπων : Νικόλαος Χατζηαργυρίου**

Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 10η Οκτωβρίου 2019.

| | | |
|---|---|---|
| ..... | ..... | ..... |
| Νικόλαος Χατζηαργυρίου<br>Καθηγητής Ε.Μ.Π. | Σταύρος Παπαθανασίου<br>Καθηγητής Ε.Μ.Π. | Παύλος Γεωργιλάκης<br>Αναπληρωτής Καθηγητής Ε.Μ.Π. |

Αθήνα,Οκτώβριος,2019

.....

Ηλίας Κάππος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

# Περίληψη

Στην παρούσα διπλωματική επιλύεται το πρόβλημα Οικονομικής Κατανομής Φορτίου με κατανεμημένο-αποκεντρωμένο τρόπο. Το πρόβλημα Οικονομικής Κατανομής Φορτίου είναι ένα από τα πιο σημαντικά προβλήματα βελτιστοποίησης στα συστήματα ηλεκτρικής ενέργειας (ΣΗΕ), όπου στόχος είναι να ελαχιστοποιηθεί το συνολικό κόστος παραγωγής των γεννητριών ενός συστήματος αλλά παράλληλα να ικανοποιείται η ζήτηση του φορτίου και να μην παραβιάζονται κάποιοι περιορισμοί σχετικοί με τη λειτουργία των γεννητριών και με το σύστημα μεταφοράς. Γενικά έχουν προταθεί πολλοί τρόποι επίλυσης του συγκεκριμένου προβλήματος, όπως η μέθοδος πολλαπλασιαστών Lagrange καθώς και η μέθοδος ADMM. Στη εν λόγω εργασία προτείνεται η επίλυση του προβλήματος Οικονομικής Κατανομής Φορτίου με τη μέθοδο ADMM, με τη διαφορά ότι ένα βήμα της (ο κεντρικός υπολογισμός) θα εκτελείται μέσω ενός "έξυπνου συμβολαίου" (smart contract) στο Blockchain, κάνοντας τον συνολικό αλγόριθμο κατανεμημένο.

Μέσω του προτεινόμενου αλγορίθμου, οι κόμβοι του δικτύου (γεννήτριες) είναι μέλη ενός Ethereum Blockchain, στο οποίο έχει αναπτυχθεί (deployed) ένα πρωτόκολλο σύμφωνα με το οποίο συλλέγονται οι λύσεις του προβλήματος βελτιστοποίησης που λύνουν τοπικά οι κόμβοι λαμβάνοντας υπόψη τα τοπικά όρια παραγωγής τους και στη συνέχεια υπολογίζεται η ανανέωση της καθολικής (global) μεταβλητής κάθε γεννήτριας, λύνοντας το πρόβλημα βελτιστοποίησης με τον περιορισμό να καλύπτεται η ζήτηση φορτίου του συστήματος. Ο κώδικας εκτελείται ταυτόχρονα σε κάθε κόμβο του δικτύου και τα αποτελέσματα είναι κοινά για όλους.

Τέλος παρατίθενται διάφορα σενάρια εκτέλεσης του παραπάνω αλγορίθμου ώστε να προκύψουν συμπεράσματα. Κάθε φορά τα αποτελέσματα συγκρίνονται με τα αποτελέσματα του αντίστοιχου κεντρικού αλγορίθμου που επιλύει το πρόβλημα Οικονομικής Κατανομής Φορτίου με την ADMM. Επίσης οι κόμβοι προκείμενου να μπορούν να στέλνουν δεδομένα στο Blockchain πληρώνουν το καύσιμο του Ethereum, το gas, και έτσι λοιπόν εξετάζεται και η οικονομική τους επιβάρυνση σε κάθε εκτέλεση του αλγορίθμου.

# Λέξεις Κλειδιά

Πρόβλημα Οικονομικής Κατανομής Φορτίου, κατανεμημένος/αποκεντρωμένος αλγόριθμος, ADMM, Blockchain, Ethereum, gas, ευφυή δίκτυα, διεσπαρμένη παραγωγή

# Abstract

The present diploma thesis addresses the Economic Dispatch Problem (EDP) in a decentralized way. The Economic Dispatch Problem (EDP) is one of the most important optimization problems in power systems, where the goal is to minimize the total cost of the power generation in a generators' system and at the same time to meet the load demand and respect the technical constraints of the generators and the transmission system constraints. Many algorithms have been proposed to solve the EDP, such as the Lagrange multipliers method and the ADMM. In this project, the ADMM method is adopted for calculating the optimal load allocation, with the exception that the centralized computation step is performed by a "smart contract" on a blockchain, leading this way to a distributed algorithm.

In the proposed algorithm, the network nodes (generators) are members of an Ethereum Blockchain, in which there is a protocol implemented that aggregates the solutions of local optimization problems taking into account their local output limits and subsequently calculates the updates of the global variable, by solving the global optimization problem integrating the total load satisfaction constraint. The code is executed simultaneously on every node of the Ethereum network providing this way common results.

Finally, various execution scenarios of the above algorithm are presented in order to draw conclusions. In each scenario, the results are compared with the results of the equivalent centralized algorithm that solves the EDP using the ADMM. Due to the fact that the computational steps in Ethereum are costly (the Ethereum fuel), the financial cost is also examined for every node in various execution scenarios.

# Key words

Economic Dispatch Problem (EDP), distributed/decentralized algorithm, ADMM, Blockchain, Ethereum, gas, smart grids, distributed generation

# Ευχαριστίες

Η παρούσα διπλωματική εργασία πραγματοποιήθηκε υπό την επίβλεψη του καθηγητή του Ε.Μ.Π. Νικόλαου Χατζηαργυρίου, τον οποίο θα ήθελα να ευχαριστήσω ιδιαιτέρως για την δυνατότητα που μου έδωσε να μελετήσω ένα τόσο ενδιαφέρον θέμα και να διευρύνω τις γνώσεις μου γύρω από αυτό.

Επίσης, ευχαριστώ τον ερευνητή κ. Ιάσονα Κουβελιώτη-Λυσίκατο, για την πολύτιμη βοήθεια που μου παρείχε κατά τη διάρκεια της διπλωματικής και τις χρήσιμες συμβουλές του.

Τέλος, τη μεγαλύτερη ευγνωμοσύνη την οφείλω στην οικογένεια μου η οποία είναι πάντα δίπλα μου τόσο στις σπουδές όσο και σε κάθε βήμα της ζωής μου, δείχνοντάς μου την αμέριστη αγάπη και υποστήριξή της.

# Περιεχόμενα

Περίληψη ..... 1
Abstract ..... 3
Ευχαριστίες ..... 5

**Κεφάλαιο 1-Εισαγωγή ..... 19**
1.1 Σκοπός και αντικείμενο της διπλωματικής εργασίας ..... 20
1.2 Δομή της διπλωματικής εργασίας ..... 21
**Κεφάλαιο 2-Συστήματα Ηλεκτρικής Ενέργειας ..... 23**
2.1 Δομή συστημάτων ηλεκτρικής ενέργειας. ..... 24
2.2 Κεντρική παραγωγή ενέργειας ..... 24
2.3 Διεσπαρμένη παραγωγή ενέργειας ..... 26
2.4 Ευφυή δίκτυα ..... 27
2.4.1 Ασύρματα δίκτυα αισθητήρων ..... 29
**Κεφάλαιο 3-Τεχνολογίες Blockchain ..... 31**
3.1 Δίκτυα υπολογιστών. ..... 32
3.2 Εισαγωγή στο Blockchain ..... 33
3.3 Χαρακτηριστικά της τεχνολογίας Blockchain ..... 34
3.3.1 Συνάρτηση κατακερματισμού ..... 35
3.3.2 Ψηφιακή υπογραφή ..... 36
3.3.3 Εξόρυξη blocks ..... 38
3.3.4 Πρωτόκολλα συναίνεσης ..... 40
3.4 Το Blockchain ως δομή δεδομένων ..... 42
3.5 Κατηγορίες Blockchain ..... 42
3.6 Ethereum ..... 43
3.6.1 Εισαγωγή στο Ethereum ..... 43
3.6.2 Λογαριασμοί ..... 43
3.6.3 Εικονική Μηχανή του Ethereum ..... 44
3.6.4 Συναλλαγές ..... 45
3.6.5 Μηνύματα ..... 46
3.6.6 Gas ..... 46

3.6.7 Εξόρυξη blocks ..... 46
3.6.8 Δομή των blocks ..... 47
.3.7 Εφαρμογή της τεχνολογίας Blockchain στον τομέα της ενέργειας ..... 47
3.7.1 Βιβλιογραφική ανασκόπηση ..... 47
**Κεφάλαιο 4-Οικονομική Κατανομή Φορτίου ..... 53**
4.1 Εισαγωγή στην Οικονομική Κατανομή Φορτίου. ..... 54
4.2 Κλασσικό μοντέλο Οικονομική Κατανομή Φορτίου ..... 55
4.3 Μέθοδος πολλαπλασιαστών Lagrange αγνοώντας τις απώλειες μεταφοράς και τα όρια παραγωγής των μονάδων ..... 60
4.3.1 Μέθοδος πολλαπλασιαστών Lagrange λαμβάνοντας υπόψη τα τεχνικά όρια των μονάδων ..... 61
**Κεφάλαιο 5-Βασικός αλγόριθμος ..... 63**
5.1 Εισαγωγή στην ADMM μέθοδο . ..... 64
5.1.1 Μέθοδος δυικής ανάβασης (Dual ascent) . ..... 64
5.1.2 Δυική αποσύνθεση (Dual decomposition) . ..... 65
5.1.3 Επαυξημένη (Augmented) συνάρτηση Lagrange και μέθοδος πολλαπλασιαστών Lagrange. ..... 66
5.1.4 ADMM. ..... 67
5.2 Αλγόριθμος που βασίζεται στη μέθοδο ADMM για τη λύση του προβλήματος Οικονομικής Κατανομής Φορτίου ..... 68
5.3 Προτεινόμενος αποκεντρωμένος αλγόριθμος ..... 74
**Κεφάλαιο 6-Εργαλεία και τεχνολογίες ..... 77**
6.1 Solidity . ..... 78
6.2 Remix ..... 78
6.3 Ganache Gui ..... 78
6.4 Python . ..... 79
6.5 Web3.py ..... 79
6.6 Άλλες βιβλιοθήκες της Python που χρησιμοποιήθηκαν ..... 80
**Κεφάλαιο 7-Ανάλυση του κώδικα της διπλωματικής ..... 81**
7.1 Ανάλυση κώδικα έξυπνου συμβολαίου . ..... 82
7.2 Ανάλυση κώδικα Python ..... 88
**Κεφάλαιο 8-Σενάρια διπλωματικής ..... 93**
8.1Βασικο σενάριο διπλωματικής- Σύστημα 6 γεννητριών ..... 94

8.2 Επίλυση του βασικού σεναρίου με χρήση του κεντρικού αλγορίθμου ..... 94
8.3 Επίλυση του βασικού σεναρίου με χρήση του αποκεντρωμένου αλγορίθμου ..... 98
8.3.1 Σύγκριση αποτελεσμάτων κεντρικού και αποκεντρωμένου αλγόριθμου .... 100
8.3.2 Κατανάλωση gas, υπόλοιπο χρηστών και έξοδα του κάθε χρήστη ..... 101
8.3.3 Εκτέλεση του αποκεντρωμένου αλγορίθμου για διαφορετικά ε ..... 104
8.3.4 Αποτελέσματα του αποκεντρωμένου αλγορίθμου με διαφορετικά ε ..... 108
8.3.5 Εκτέλεση του αποκεντρωμένου αλγορίθμου για διαφορετικά ρ ..... 110
8.3.6 Αποτελέσματα του αποκεντρωμένου αλγορίθμου με διαφορετικά ρ ..... 115
8.4 Συστήματα περισσότερων γεννητριών ..... 116
8.4.1 Σύστημα 10 γεννητριών ..... 116
8.4.2 Σύστημα 20 γεννητριών ..... 118
8.4.3 Σύστημα 50 γεννητριών ..... 119
8.4.4 Αποτελέσματα του αποκεντρωμένου αλγορίθμου για διαφορετικό πλήθος γεννητριών ..... 119
**Κεφάλαιο 9-Συμπεράσματα και προοπτικές ..... 123**
9.1 Συμπεράσματα . ..... 124
9.2 Προοπτικές για περεταίρω έρευνα ..... 126
Βιβλιογραφία ..... 127
Παράρτημα ..... 133

# Κατάλογος Εικόνων

Εικόνα 2.1 Κεντρική παραγωγή ενέργειας ..... 25

Εικόνα 2.2 Διεσπαρμένη παραγωγή ενέργειας ..... 26

Εικόνα 2.3 Το ευφυές δίκτυο σύμφωνα με το μοντέλο του Nist ..... 28

Εικόνα 3.1 Κεντρικά, Αποκεντρωμένα και Κατανεμημένα δίκτυα ..... 32

Εικόνα 3.2 Απλοποιημένη αναπαράσταση της δομής ενός Blockchain (txn=transaction) ..... 34

Εικόνα 3.3 Συνάρτηση κατακερματισμού ..... 35

Εικόνα 3.4 Ψηφιακή υπογραφή (φάση υπογραφής και φάση επικύρωσης) ..... 36

Εικόνα 3.5 Συναλλαγή στο bitcoin ..... 37

Εικόνα 3.6 Διαφορετικές αλυσίδες από blocks (το πράσινο block είναι το genesis block). Η αλυσίδα που αποτελείται από τα μαύρα blocks είναι η κανονική μιας και έχει μεγαλύτερο μήκος ..... 40

Εικόνα 3.7 Οι συναλλαγές ενός block αποθηκευμένες σε δέντρο Merkle. ..... 42

Εικόνα 3.8 Δέντρο κατάστασης (State tree) ..... 47

Εικόνα 6.1 Λειτουργία της web3.py ..... 69

# Κατάλογος Διαγραμμάτων

Διάγραμμα 4.1 Καμπύλη λειτουργικού κόστους της γεννήτριας *i* σε συνάρτηση με την ενεργό παραγωγής της ..... 56

Διάγραμμα 4.2 Οριακό κόστος λειτουργίας της γεννήτριας *i* συναρτήσει της ενεργού παραγωγής της ..... 57

Διάγραμμα 4.3 Κατανομή της ισχύος σε ένα σύστημα με 2 γεννήτριες ..... 59

Διάγραμμα 8.1 Η εξέλιξη της ισχύος εξόδου κάθε γεννήτριας κατά τη διάρκεια του κεντρικού αλγορίθμου (για σύστημα 6 γεννητριών) ..... 95

Διάγραμμα 8.2 Η εξέλιξη της primal residual της πρώτης γεννήτριας κατά τη διάρκεια του κεντρικού αλγορίθμου (για σύστημα 6 γεννητριών)..... 96

Διάγραμμα 8.3 Η εξέλιξη της dual residual της πρώτης γεννήτριας κατά τη διάρκεια του κεντρικού αλγορίθμου (για σύστημα 6 γεννητριών)..... 96

Διάγραμμα 8.4 Η εξέλιξη της ισορροπίας μεταξύ παραγωγής και ζήτησης κατά τη διάρκεια εκτέλεσης του κεντρικού αλγορίθμου (για σύστημα 6 γεννητριών) ..... 97

Διάγραμμα 8.5 Μεγέθυνση του διαγράμματος 8.4..... 97

Διάγραμμα 8.6 Η εξέλιξη της ενεργού ισχύος εξόδου κάθε γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών)..... 98

Διάγραμμα 8.7 Η εξέλιξη της primal residual της πρώτης γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών) ..... 99

Διάγραμμα 8.8 Η εξέλιξη της dual residual της πρώτης γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών). ..... 99

Διάγραμμα 8.9 Η εξέλιξη της ισορροπίας μεταξύ παραγωγής και ζήτησης κατά τη διάρκεια εκτέλεσης του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών) ..... 100

Διάγραμμα 8.10 Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ε=0)..... 101

Διάγραμμα 8.11 Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ε=10) ..... 105

Διάγραμμα 8.12 Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ε=20)..... 106

Διάγραμμα 8.13 Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ε=50)..... 107

Διάγραμμα 8.14 Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ε=100 ) ..... 108

Διάγραμμα 8.15 Αριθμός των επαναλήψεων που απαιτούνται για σύγκλιση σε συνάρτηση με το ε ..... 109

Διάγραμμα 8.16 Μέση κατανάλωση gas σε συνάρτηση με το ε ..... 109

Διάγραμμα 8.17 Μέση χρηματική κατανάλωση (€) σε συνάρτηση με το ε ..... 109

Διάγραμμα 8.18 Μέσο σφάλμα σε σχέση με τον κεντρικό αλγόριθμο συναρτήσει του ε ..... 110

Διάγραμμα 8.19 Σύγκριση αποκεντρωμένου αλγορίθμου με $\rho = 0{,}1$ με τον κεντρικό αλγόριθμο (για σύστημα 6 γεννητριών) ..... 111

Διάγραμμα 8.20 Η εξέλιξη της ισχύος εξόδου κάθε γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου με ρ=0,1 (για σύστημα 6 γεννητριών) ..... 111

Διάγραμμα 8.21 Η εξέλιξη της primal residual της γεννήτριας 1 (για σύστημα 6 γεννητριών και ρ=0,1) ..... 112

Διάγραμμα 8.22 Η εξέλιξη της dual residual της γεννήτριας 1 (για σύστημα 6 γεννητριών και ρ=0,1) ..... 112

Διάγραμμα 8.23 Ισορροπία παραγωγής-ζήτησης κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ρ=0,1) ..... 113

Διάγραμμα 8.24 Σύγκριση αποκεντρωμένου αλγορίθμου με τον κεντρικό αλγόριθμο (για σύστημα 6 γεννητριών και ρ=5) ..... 114

Διάγραμμα 8.25 Η εξέλιξη της ισχύος εξόδου κάθε γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ρ=5) ..... 114

Διάγραμμα 8.26 Αριθμός επαναλήψεων του αποκεντρωμένου αλγορίθμου συναρτήσει του $\rho$ ..... 115

Διάγραμμα 8.27 Μέση κατανάλωση gas συναρτήσει του $\rho$ ..... 115

Διάγραμμα 8.28 Μέση χρηματική κατανάλωση σε € συναρτήσει του $\rho$ ..... 116

Διάγραμμα 8.29 Μέσο σφάλμα συναρτήσει του $\rho$ ..... 116

Διάγραμμα 8.30 Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου για σύστημα 10 γεννητριών ..... 117

Διάγραμμα 8.31 Η εξέλιξη της ισχύος κάθε γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 10 γεννητριών) ..... 117

Διάγραμμα 8.32 Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου για σύστημα 20 γεννητριών ..... 118

Διάγραμμα 8.33 Εξέλιξη ισχύος εξόδου κάθε γεννήτριας για σύστημα 20 γεννητριών ..... 118

Διάγραμμα 8.34: Εξέλιξη ισχύος εξόδου κάθε γεννήτριας για σύστημα 50 γεννητριών ..... 119

Διάγραμμα 8.35 Αριθμός επαναλήψεων που απαιτούνται για σύγκλιση συναρτήσει του αριθμού των γεννητριών ..... 120

Διάγραμμα 8.36 Μέση κατανάλωση gas συναρτήσει του αριθμού των γεννητριών 120

Διάγραμμα 8.37 Μέση χρηματική κατανάλωση (€) συναρτήσει του αριθμού των γεννητριών ..... 120

# Κατάλογος Πινάκων

Πίνακας 8.1 Παράμετροι των 6 γεννητριών του συστήματος ..... 94

Πίνακας 8.2 Ισχύς εξόδου της κάθε γεννήτριας μετά το πέρας του κεντρικού αλγορίθμου (για σύστημα 6 γεννητριών) ..... 95

Πίνακας 8.3 Ισχύς εξόδου της κάθε γεννήτριας μετά το πέρας του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών) ..... 98

Πίνακας 8.4 Υπόλοιπο λογαριασμού κάθε κόμβου, κατανάλωση gas και ether (για σύστημα 6 γεννητριών και gas price=20gWei/gas) ..... 102

Πίνακας 8.5 Ποσό που ξόδεψε κάθε κόμβος σε € (για σύστημα 6 γεννητριών και gas price=20gWei/gas) ..... 103

Πίνακας 8.6 Υπόλοιπο λογαριασμού κάθε κόμβου και ποσό που σπαταλήθηκε (σε ether και €) (για σύστημα 6 γεννητριών και gas price=3gWei/gas) ..... 104

Πίνακας 8.7 Κατανάλωση gas κάθε κόμβου και ποσό που σπαταλήθηκε σε ether και € με ε=10 (για σύστημα 6 γεννητριών) ..... 106

# Κεφάλαιο 1

## *Εισαγωγή*

## 1.1 Σκοπός και αντικείμενο της διπλωματικής εργασίας

Η ηλεκτρική ενέργεια είναι βασικό κομμάτι της ζωής του ανθρώπου. Σχεδόν σε κάθε δραστηριότητα της ζωής μας χρησιμοποιούμε ηλεκτρική ενέργεια. Τα τελευταία χρόνια λόγω της μεγάλης περιβαλλοντικής μόλυνσης που έχει ως συνέπεια την όξυνση του φαινομένου του θερμοκηπίου, τη διόγκωση της τρύπας του όζοντος, το λιώσιμο των πάγων κ.α. ερευνώνται τρόποι για παραγωγή ενέργειας από "καθαρές" πηγές. Προς αυτή τη κατεύθυνση οδηγεί και η μείωση των κοιτασμάτων των ορυκτών καυσίμων. Έτσι η παραγωγή ενέργειας από ανανεώσιμες πηγές έχει γίνει επιτακτική και αναγκαία.

Τα τελευταία χρόνια παρατηρείται μεγάλη διείσδυση ΑΠΕ στο δίκτυο ηλεκτρικής ενέργειας. Είναι λοιπόν ιδιαίτερα κρίσιμο, η λειτουργία τους σε συνδυασμό με τους ήδη υπάρχοντες σταθμούς παραγωγής να είναι ομαλή χωρίς να επηρεάζει την ευστάθεια του δικτύου. Λόγω της μη σταθερής παραγωγής των ΑΠΕ καθώς και το γεγονότος ότι συνήθως είναι διασκορπισμένα μέσα στο δίκτυο, πρέπει να υπάρχει η κατάλληλη εποπτεία και διαχείριση του δικτύου ώστε να εξασφαλίζεται η αποδοτικότητα του.

Πέρα από τη διείσδυση των ΑΠΕ στο δίκτυο και άλλα γεγονότα δείχνουν ότι το δίκτυο ηλεκτρικής ενέργειας αρχίζει και εξελίσσεται, κατευθυνόμενο προς μία πιο ενεργητική μορφή. Ήδη ο απλός πολίτης έχει αρχίσει να συμμετέχει στην παραγωγή ηλεκτρικής ενέργειας, "έξυπνες" συσκευές αρχίζουν να χρησιμοποιούνται κατά κόρον, όλο και περισσότερα ηλεκτρικά οχήματα εμφανίζονται και άμεση προβλέπεται η αντικατάσταση των κλασσικών μετρητών ηλεκτρικής ενέργειας με άλλους πιο "έξυπνους". Έτσι λοιπόν τίθεται το θέμα της διαχείρισης όλων αυτών των οντοτήτων που θα "ζουν" στο δίκτυο. Η κεντρική μορφή διαχείρισης που υπάρχει αυτή τη στιγμή δείχνει ότι μάλλον δεν είναι η κατάλληλη και άλλες πιο αποκεντρωμένες μέθοδοι ερευνώνται.

Μια άλλη τεχνολογία για την οποία γίνεται λόγος τα τελευταία χρόνια είναι αυτή των blockchain. Ο πιο γνωστός τομέας που χρησιμοποιούνται περισσότερο μέχρι σήμερα είναι αυτός των κρυπτονομισμάτων και ιδιαίτερα το bitcoin. Το bitcoin επιδιώκει να καταργήσει τους ενδιάμεσους φορείς οι οποίοι διαχειρίζονται τις ηλεκτρονικές οικονομικές συναλλαγές και να δημιουργήσει ένα ισχυρό αποκεντρωμένο δίκτυο όπου η επιβεβαίωση για τις συναλλαγές θα γίνεται από το ίδιο το δίκτυο. Το γεγονός αυτό έχει οδηγήσει στο να ερευνάται η χρήση της τεχνολογίας του blockchain όχι μόνο στο τομέα των οικονομικών συναλλαγών αλλά και σε πολλούς ακόμα, όπως στον τομέα της ενέργειας, της εφοδιαστικής αλυσίδας, σε ψηφοφορίες, στον τομέα της αυτοκινητοβιομηχανίας κ.α.

Αυτός είναι και ο σκοπός της διπλωματικής. Να ερευνηθεί η χρήση του blockchain σε μία μέθοδο βελτιστοποίησης ενός γνωστού προβλήματος βελτιστοποίησης των ΣΗΕ, του προβλήματος οικονομικής κατανομής φορτίου. Αρχικά αφού γίνει μία σύντομη εισαγωγή περί εξέλιξης των δικτύων ηλεκτρικής ενέργειας καθώς και μία παρουσία-

ση της τεχνολογίας των blockchains θα εξηγηθεί το πρόβλημα οικονομικής κατανομής φορτίου αλλά και κάποιες συνηθισμένες μέθοδοι που χρησιμοποιούνται για την επίλυση του. Στη συνέχεια θα παρουσιαστεί ένας αλγόριθμος επίλυσης του προβλήματος με τη μέθοδο ADMM και θα αναπτυχθεί ένας άλλος αλγόριθμος που χρησιμοποιεί το blockchain για την υλοποίηση κάποιων βημάτων της μεθόδου ADMM, έτσι ώστε να λύνεται το πρόβλημα αποκεντρωμένα. Τέλος θα εφαρμοστεί ο αλγόριθμος σε διαφορετικές περιπτώσεις συστημάτων ώστε να εξαχθούν κάποια συμπεράσματα.

## 1.2 Δομή της διπλωματικής εργασίας

Ακολουθεί μια συνοπτική παρουσίαση των κεφαλαίων της παρούσας διπλωματικής:

**Κεφάλαιο 1$^{ο}$ : Εισαγωγή**

Στο κεφάλαιο αυτό αναλύεται ο σκοπός και το περιεχόμενο της διπλωματικής εργασίας καθώς και το πρόβλημα το οποίο καλείται να λύσει.

**Κεφάλαιο 2$^{ο}$ : Συστήματα Ηλεκτρικής Ενέργειας**

Στο εν λόγω κεφάλαιο παρουσιάζονται συνοπτικά κάποιες βασικές αρχές που διέπουν τα δίκτυα ηλεκτρικής ενέργειας καθώς και όροι όπως διεσπαρμένη παραγωγή, έξυπνο δίκτυο και δίκτυα αισθητήρων. Γενικά οι έννοιες που αναλύονται είναι άρρηκτα συνδεδεμένες με το κύριο αντικείμενο της διπλωματικής εργασίας.

**Κεφάλαιο 3$^{ο}$ : Τεχνολογίες Blockchain**

Στο παρόν κεφάλαιο παρουσιάζεται η τεχνολογία του blockchain. Συγκεκριμένα αναλύονται κάποιες βασικές αρχές που διέπουν την λειτουργία του (όπως η ψηφιακή υπογραφή και τα πρωτόκολλα συναίνεσης). Επίσης περιγράφεται και μια βασική πλατφόρμα που χρησιμοποιεί το blockchain, το Ethereum και γίνεται και βιβλιογραφική ανασκόπηση με εργασίες που πραγματεύονται εφαρμογές blockchain στον ενεργειακό τομέα.

**Κεφάλαιο 4$^{ο}$ : Οικονομική Κατανομή Φορτίου**

Στο κεφάλαιο αυτό γίνεται μια παρουσίαση του προβλήματος της οικονομικής κατανομής φορτίου και περιγράφονται δύο από τις πιο γνωστές μεθόδους επίλυσής του

**Κεφάλαιο 5ο: Βασικός Αλγόριθμος**

Στο παρόν κεφάλαιο αναλύεται ένας αλγόριθμος για την επίλυση του προβλήματος οικονομικής κατανομής φορτίου με τη μέθοδο ADMM. Επίσης αναπτύσσεται και ένας συνδυασμός αυτού του αλγορίθμου με την τεχνολογία blockchain, ώστε να δημιουργηθεί ένας αποκεντρωμένος αλγόριθμος.

**Κεφάλαιο 6$^{ο}$: Εργαλεία και τεχνολογίες**

Στο κεφάλαιο αυτό περιγράφονται συνοπτικά τα μέσα που χρησιμοποιήθηκαν για την υλοποίηση του αποκεντρωμένου αλγορίθμου.

**Κεφάλαιο 7$^{ο}$ : Ανάλυση του κώδικα της διπλωματικής**

Στο παρόν κεφάλαιο αναλύονται ο κώδικες που συντάχθηκαν στο πλαίσιο αυτής της διπλωματικής.

**Κεφάλαιο 8$^{ο}$ : Σενάρια διπλωματικής**

Στο συγκεκριμένο κεφάλαιο εφαρμόζεται ο αλγόριθμος που αναπτύχθηκε σε ένα δίκτυο 6 γεννητριών και μεταβάλλοντας κάποιες παραμέτρους του αλγορίθμου καταγράφονται κάποια αποτελέσματα. Επίσης εφαρμόζεται ο αλγόριθμος και σε δίκτυα περισσότερων γεννητριών

**Κεφάλαιο 9$^{ο}$ : Συμπεράσματα και προοπτικές**

Στο τελευταίο κεφάλαιο της διπλωματικής καταγράφονται τα συμπεράσματα τα οποία μπορούν να εξαχθούν από τα αποτελέσματα που προέκυψαν καθώς και κάποιες μελλοντικές προοπτικές έρευνας που προκύπτουν από την παρούσα μελέτη.

# Κεφάλαιο 2

## *Συστήματα Ηλεκτρικής Ενέργειας*

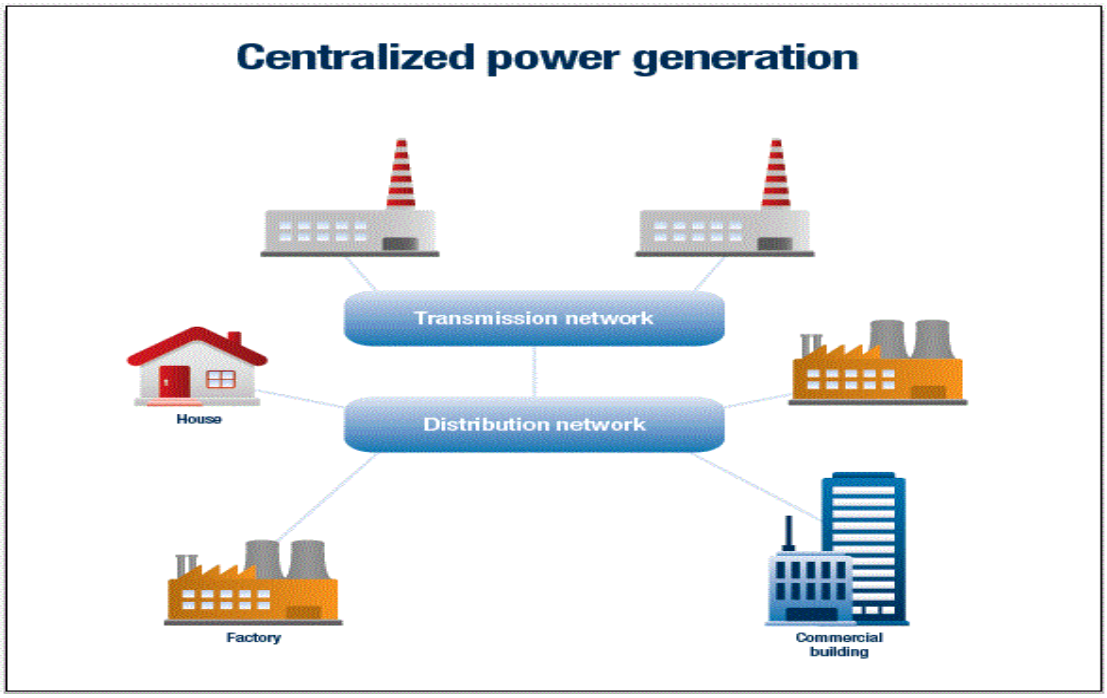
## 2.1 Δομή συστημάτων ηλεκτρικής ενέργειας

Σύστημα ηλεκτρικής ενέργειας είναι το σύνολο των εγκαταστάσεων και των μέσων που χρησιμοποιούνται προκειμένου η ηλεκτρική ενέργεια να φτάσει στις εξυπηρετούμενες περιοχές κατανάλωσης. Ένα σύστημα ηλεκτρικής ενέργειας προκειμένου να λειτουργεί καλά θα πρέπει να μπορεί να παρέχει ηλεκτρική ενέργεια οπουδήποτε υπάρχει ζήτηση με το ελάχιστο δυνατό κόστος, να διατηρεί τη συχνότητα και τη τάση σταθερή, να εξασφαλίζει υψηλή αξιοπιστία τροφοδότησης αλλά και συνάμα να μην επιβαρύνει το περιβάλλον. Προκειμένου η ηλεκτρική ενέργεια να φτάσει στον τελικό καταναλωτή θα πρέπει πρώτα να περάσει από τρία στάδια: την παραγωγή, τη μεταφορά και τέλος τη διανομή. Σημαντικό επίσης είναι η ηλεκτρική ενέργεια να παράγεται την στιγμή που καταναλώνεται μιας και η ροής της είναι συνεχής και επίσης η αποθήκευσή της είναι δύσκολη και ακριβή.

Παραγωγή ηλεκτρικής ενέργειας είναι η διαδικασία με την οποία μία μορφή πρωτογενούς ενέργειας μετατρέπεται σε ηλεκτρική. Συνήθως η πρωτογενής ενέργεια πρώτα μετατρέπεται σε μηχανική μέσω κινητήρων (π.χ. στροβίλων) και στη συνέχεια μέσω των γεννητριών μετατρέπεται σε ηλεκτρική. Το μεγαλύτερο μέρος της παγκόσμιας ηλεκτροπαραγωγής γίνεται στους θερμοηλεκτρικούς σταθμούς, όπου εκμεταλλευόμαστε την χημική ενέργεια ορυκτών καυσίμων (π.χ. λιγνίτης, άνθρακας, πετρέλαιο). Επίσης μεγάλο μέρος ηλεκτρικής ενέργειας παράγεται στους υδροηλεκτρικούς σταθμούς, όπου εκμεταλλευόμαστε την πτώση των υδάτων, στους πυρηνικούς σταθμούς με πυρηνική σχάση (π.χ. ουράνιο, πλουτώνιο) και τα τελευταία χρόνια γίνεται μια προσπάθεια για παραγωγή από ανανεώσιμες πηγές ενέργειας όπως η ηλιακή, η αιολική, η γεωθερμία κ.α. Η ηλεκτρική ενέργεια μιας και παράγεται σε μεγάλους σταθμούς οι οπαίοι είναι απομακρυσμένοι από τα κέντρα κατανάλωσης θα πρέπει στη συνέχεια με μεταφερθεί. Η μεταφορά της ηλεκτρικής ενέργειας γίνεται μέσω γραμμών μεταφοράς υψηλής ή υπερύψηλης τάσης. Η υψηλή τάση χρησιμοποιείται προκειμένου να ελαχιστοποιούνται οι απώλειες κατά τη μεταφορά. Έτσι λοιπόν η ενέργεια μεταφέρεται στους υποσταθμούς από τους οποίους θα γίνει η διανομή ηλεκτρικής ενέργειας στους τελικούς καταναλωτές [1].

## 2.2 Κεντρική παραγωγή ενέργειας

Όπως αναλύσαμε και παραπάνω η δομή των συστημάτων ηλεκτρικής ενέργειας περιλαμβάνει κατά βάση μεγάλους σταθμούς παραγωγής, οι οποίοι παράγουν μεγάλες ποσότητες ηλεκτρικής ενέργειας, η οποία μεταφέρεται και διανέμεται στη συνέχεια σε απομακρυσμένα κέντρα κατανάλωσης. Αυτό είναι το λεγόμενο κεντρικό σύστημα παραγωγής ηλεκτρικής ενέργειας (centralized energy generation) [2].




Εικόνα 2.1: Κεντρική παραγωγή ενέργειας [3]

Το βασικό πλεονέκτημα αυτού του συστήματος είναι ότι λόγω των μεγάλων σύγχρονων γεννητριών από τις οποίες αποτελούνται (ρυθμιστές τάσης και συχνότητας) επιτυγχάνεται ένα καλό επίπεδο ευστάθειας στο σύστημα [4].

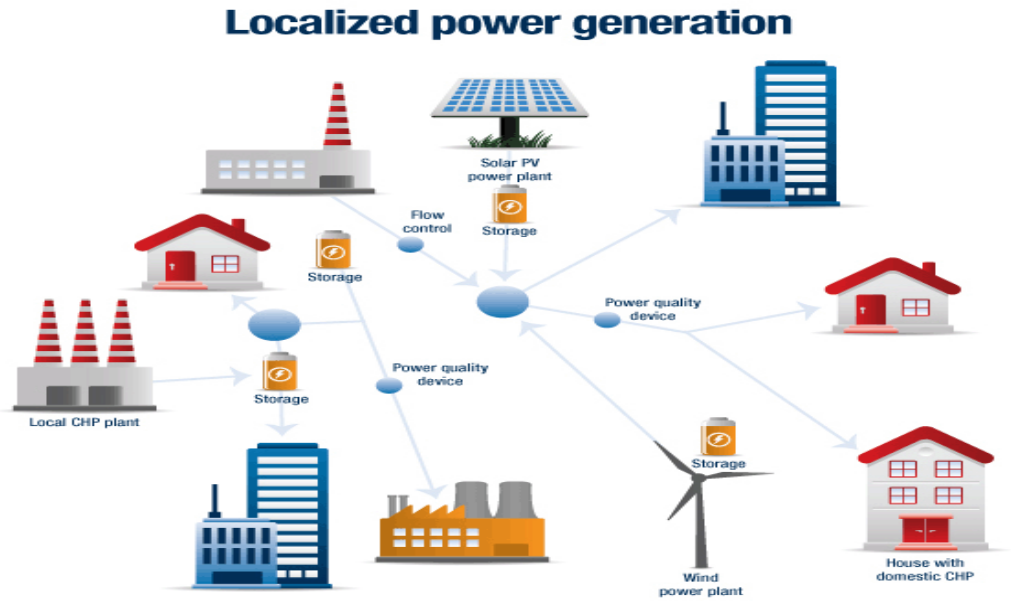
Όμως η κεντρική παραγωγή ενέργειας έχει κάποια αρνητικά μεταξύ των οποίων συγκαταλέγονται :

- Το μεγαλύτερο μέρος της ηλεκτρικής ενέργειας που παράγεται στους κεντρικούς σταθμούς παραγωγής προέρχεται από την καύση ορυκτών καυσίμων. Το αποτέλεσμα είναι μεγάλη περιβαλλοντική ρύπανση (φαινόμενο θερμοκηπίου, τρύπα του όζοντος) [5].
- Στο κεντρικό μοντέλο παραγωγής οι σταθμοί είναι απομακρυσμένοι από τα κέντρα κατανάλωσης με αποτέλεσμα να χρειάζεται ένα μεγάλο σύστημα μεταφορά και διανομής έτσι ώστε η ενέργεια να φτάσει στον καταναλωτή. Η συντήρηση του δικτύου αλλά και η επέκτασή του όταν απαιτείται είναι μια διαδικασία ιδιαίτερα δαπανηρή και δύσκολη. Επίσης λόγω των ιδιαίτερα μεγάλων αποστάσεων είναι λογικό οι θερμικές απώλειες να είναι μεγάλες.
- Η ηλεκτροπαραγωγή συχνά ελέγχεται από μεγάλες εταιρίες ηλεκτρισμού οι οποίες λόγω του ότι έχουν το προνόμιο του μονοπωλίου συχνά τιμολογούν ιδιαίτερα ακριβά και χωρίς να παρέχουν αξιόπιστες υπηρεσίες.

Όλα τα παραπάνω σε συνδυασμό με την ανάπτυξη τεχνολογιών ΑΠΕ καθώς και το γεγονός ότι πλέον ο κάθε πολίτης μπορεί να παράγει μόνος του ηλεκτρική ενέργεια έχουν οδηγήσει στην αμφισβήτηση του τωρινού κεντρικού συστήματος και στην ανεύρεση άλλων λύσεων.

## 2.3 Διεσπαρμένη παραγωγή ενέργειας

Τα τελευταία χρόνια διερευνάται ιδιαίτερα η παραγωγή ενέργειας κοντά στο φορτίο. Διεσπαρμένη παραγωγή ορίζεται ως η παραγωγή ενέργειας μικρής κλίμακας, με τιμές που κατά κανόνα κυμαίνονται από 1kW έως 100 MW κοντά στο σημείο κατανάλωσης. Μονάδες διεσπαρμένης παραγωγής (DERs - Distributed Energy Resources) μπορούν να θεωρηθούν οι ανανεώσιμες πηγές ενέργειας, μικροτουρμπίνες, κυψέλες καυσίμου, μηχανές εσωτερικές καύσης κ.α.[5] .




Εικόνα 2.2: Διεσπαρμένη παραγωγή ενέργειας [4]

Τα πλεονεκτήματα της διεσπαρμένης παραγωγής είναι μεταξύ άλλων [6],[7],[8] :

- Η ύπαρξη εγκατεστημένης ισχύος για παροχή εφεδρείας σε περίπτωση διακοπής της τροφοδοσίας ή και για ψαλιδισμό αιχμών του καταναλωτή και τη μείωση της κατανάλωσης ενέργειας, με τα αντίστοιχα οικονομικά οφέλη.
- Μέσω της συμπαραγωγής ηλεκτρισμού και θερμότητας τα οφέλη είναι τεράστια λόγω της εξοικονόμησης ενέργειας και χρημάτων .
- Λόγω ότι οι περισσότεροι διεσπαρμένοι πόροι είναι κυρίως ΑΠΕ, έχουμε παραγωγή καθαρής ενέργειας και απεξάρτηση από συμβατικούς ενεργειακούς πόρους οι οποίοι αρχίζουν και εξαντλούνται .
- Αυξημένη αξιοπιστία στην παρεχόμενη ενέργεια και στην ποιότητα ισχύος. Λόγω της μεγάλης βελτίωσης των συστημάτων ελέγχου και αυτοματισμού οι μονάδες διεσπαρμένης παραγωγής σε περιόδους όπου υπάρχει σφάλμα στο βασικό δίκτυο θα μπορούν να τροφοδοτήσουν τους τοπικούς καταναλωτές δημιουργώντας τοπικές νησίδες (μικροδίκτυα) .
- Δίνεται η δυνατότητα στον ιδιώτη να συμμετέχει στην παραγωγή ηλεκτρικής ενέργειας και επομένως απελευθερώνεται η αγορά.

- Λόγω της συνεχούς αύξησης της ζήτησης ηλεκτρικής ενέργειας, απαιτείται επέκταση του δικτύου με την κατασκευή νέων γραμμών μεταφοράς. Με χρήση μονάδων διεσπαρμένης παραγωγής μπορούμε να τροφοδοτήσουμε το φορτίο τοπικά και επομένως να αποφύγουμε την κατασκευή νέων γραμμών και άρα να έχουμε και μικρότερες απώλειες.

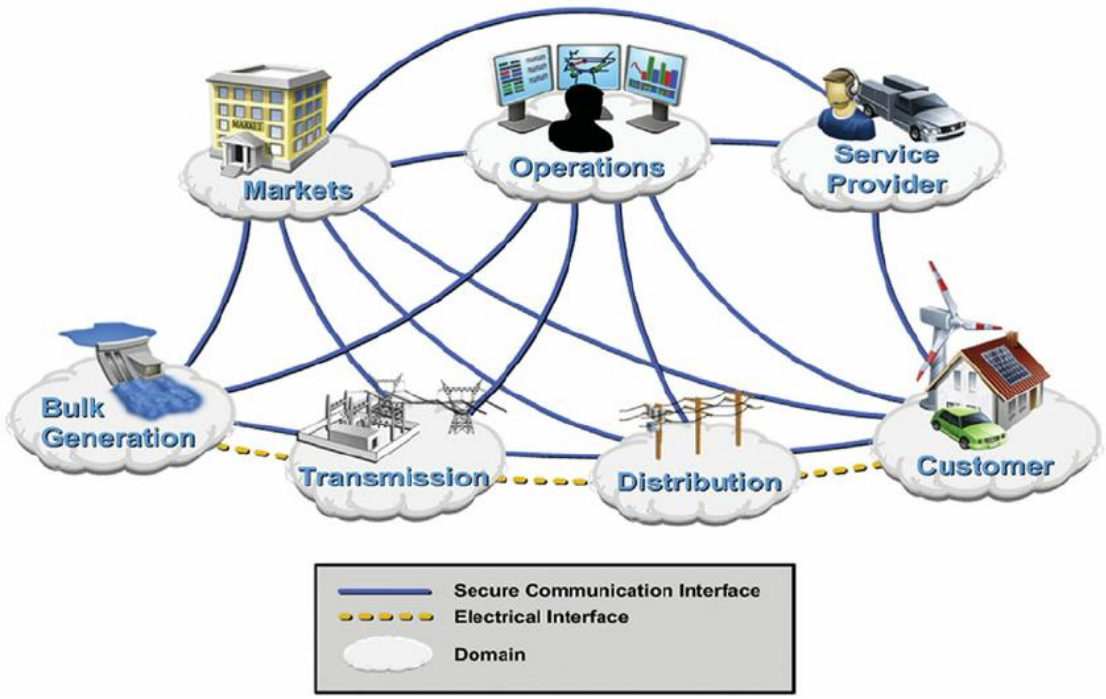
Πέρα από τα παραπάνω πλεονεκτήματα που έχουν οι μονάδες διεσπαρμένης παραγωγής, δημιουργούν και κάποια προβλήματα τα οποία δρουν ως ανασταλτικός παράγοντας για τη μεγαλύτερη διείσδυσή τους στο δίκτυο ενέργειας. Κάποια από αυτά είναι τα ακόλουθα [5],[6],[7],[9] :

- Λόγω της μεγάλης στοχαστικότητας των ΑΠΕ, θα μπορεί να καλύπτεται μόνο συγκριμένο ποσοστό της ζήτησης και θα πρέπει να τηρείται εφεδρεία από συμβατικές μονάδες παραγωγής.
- Αρκετά τεχνικά ζητήματα όπως:
  1) αυξημένες αρμονικές ρεύματος και τάσης λόγω των μετατροπέων ισχύος που χρησιμοποιούνται
  2) πιθανή αντιστροφή ισχύος από τη χαμηλή στη μέση τάση και έτσι θα απαιτούνται νέοι διακόπτες και στα δύο επίπεδα τάσης
  3) η μεταβολή των χαρακτηριστικών των βραχυκυκλωμάτων του δικτύου και η δημιουργία υπερτάσεων και υποτάσεων κατά την είσοδο και έξοδο των μονάδων
- Ένα επίσης σημαντικό ζήτημα είναι το υψηλό κόστος κεφαλαίου ανά kW εγκατεστημένης ισχύος συγκριτικά με τους μεγάλους κεντρικούς σταθμούς παραγωγής. Διαφορές όμως υπάρχουν και στα κόστη κεφαλαίου για διαφορετικές τεχνολογίες διεσπαρμένης παραγωγής και μπορούν να ποικίλλουν από 1000 €/kW έως 20000 €/kW στις τουρμπίνες καύσης και τις κυψέλες καυσίμου αντίστοιχα.
- Κάποια ρυθμιστικά θέματα για την αγορά ενέργειας όταν περιλαμβάνεται διεσπαρμένη παραγωγή τα οποία έχουν να κάνουν με την αποζημίωση της παρεχόμενης ενέργειας αλλά και τις επικουρικές υπηρεσίες που οι μονάδες διεσπαρμένης παραγωγής προσφέρουν.

## 2.4 Ευφυή δίκτυα

Ευφυές δίκτυο (smart grid) ονομάζεται το δίκτυο που ενσωματώνει τεχνολογία πληροφοριών και επικοινωνιών (Information Communication Technology – ICT) σε όλες τις πτυχές της παραγωγής, τη μεταφοράς και της κατανάλωσης ηλεκτρικής ενέργειας, προκειμένου να ελαχιστοποιηθούν οι περιβαλλοντικές επιπτώσεις, να βελτιωθούν οι αγορές, να βελτιωθεί η αξιοπιστία και η εξυπηρέτηση, να μειωθούν οι δαπάνες και να βελτιωθεί η αποδοτικότητα [10]. Είναι ένα δίκτυο υποδομών παραγωγής, μεταφοράς και διανομής, στο όποιο υπάρχουν αισθητήρες και υπολογιστές, που παρακολουθούν και διαχειρίζονται την ροή ενέργειας[11]

Το έξυπνο δίκτυο μπορεί να περιγραφεί σαν ένα σύνολο οντοτήτων που επικοινωνούν μεταξύ τους. Ο τρόπος αυτός απεικόνισης του έξυπνου δικτύου προτάθηκε από τον οργανισμό NIST και χωρίζει το έξυπνο δίκτυο σε επτά συνεργαζόμενους τομείς καθένας από τους οποίους περιλαμβάνει μια ή περισσότερες συσκευές-οντότητες, συστήματα ή προγράμματα (π.χ. έξυπνοι μετρητές, συστήματα αποθήκευσης ενέργειας), τα οποία ανταλλάσσουν πληροφορίες και παίρνουν αποφάσεις με σκοπό την εύρυθμη λειτουργία του δικτύου [12]. Αυτά φαίνονται και στο εικόνα 2.3 που ακολουθεί :




Εικόνα 2.3: Το ευφυές δίκτυο σύμφωνα με το μοντέλο του Nist [12]

Από τα παραπάνω συμπεραίνουμε ότι το έξυπνο δίκτυο είναι ουσιαστικά η εξέλιξη του σημερινού δικτύου. Βασικοί πυλώνες του είναι [5],[8]:

- Η ευφυής συνύπαρξη κεντρικής και διεσπαρμένης παραγωγής με σκοπό τη μείωση του άνθρακα και τον αποδοτικότερο χειρισμό της ζήτησης.
- Ο πολίτης να συμμετέχει ενεργά στη διαδικασία παραγωγής ενέργειας (prosumer = consumer + producer).
- Χρονομεταβλητή τιμολόγηση ενέργειας και διάφορα κίνητρα εξαρτώμενα από το μεταβαλλόμενο φορτίο.

Μερικά από τα πλεονεκτήματα ενός ευφυούς δικτύου είναι [12],[13],[14],[15]:

- Η χρησιμοποίηση ΑΠΕ θα έχει πολλά περιβαλλοντικά οφέλη και επομένως αειφόρο ανάπτυξη (sustainability) .
- Όπως αναφέραμε και παραπάνω τα έξυπνα δίκτυα απαρτίζονται από αισθητήρες. Αυτό δίνει τη δυνατότητα στο δίκτυο να προβλέψει τυχόν σφάλματα που πρόκειται να συμβούν ή να ανιχνεύσει ήδη υπάρχοντα. Και όλα αυτά χωρίς την ανάγκη ανθρώπινης εποπτείας. Έτσι λοιπόν το έξυπνο δίκτυο έχει τη δυ-

νατότητα αυτοΐασης (self-healing network), μειώνοντας τον κίνδυνο για κατάρρευση του συστήματος (blackout). Ουσιαστικά ενισχύεται ο βαθμός της αδιάλειπτης παροχής ηλεκτρικής ενέργειας.

- Μέσω των έξυπνων μετρητών θα μπορούν να συλλέγονται δεδομένα τα οποία θα βοηθούν στην καλύτερη πρόβλεψη φορτίου, άρα και σε λιγότερες διακοπές ρεύματος .
- Δίνει τη δυνατότητα στον καταναλωτή να ελέγχει σε πραγματικό χρόνο την ενεργειακή του κατανάλωση και να μεταθέτει τη λειτουργία των συσκευών του σε ώρες της μέρας που η ηλεκτρική ενέργεια είναι φθηνότερη. Ο διαχειριστής από την πλευρά του θα μπορεί να ενημερώνει τον καταναλωτή για αιχμές φορτίου μέσα στη διάρκεια της μέρας, και να του κάνει αίτημα για μετατόπιση των φορτίων του και αν ανταποκριθεί στο αίτημα του να έχει κάποια επιβράβευση (π.χ. έκπτωση στο μηνιαίο λογαριασμό).
- Αξιοποίηση της τεχνολογίας των ηλεκτρικών οχημάτων. Τα ηλεκτρικά οχήματα δεν θα αποτελούν απλά ένα φορτίο. Μέσω των μπαταριών τους θα μπορούν συνδεόμενα στο δίκτυο να παρέχουν ηλεκτρική ενέργεια έτσι ώστε να εξισορροπούν τη διακοπτόμενη παραγωγή από ΑΠΕ ή σε περίπτωση που η παραγωγή ΑΠΕ είναι μεγαλύτερη της ζήτησης θα μπορούν να απορροφούν αυτή τη περίσσεια ενέργειας και να αποτρέπουν την απόρριψη της. Έτσι τα ηλεκτρικά οχήματα μπορούν να παρέχουν επικουρικές υπηρεσίες (ancillary services), συνδράμοντας στη σταθερότητα του δικτύου. Αυτό είναι το λεγόμενο μοντέλο Vehicle-to-Grid.

### 2.4.1 Ασύρματα δίκτυα αισθητήρων

Ένα ασύρματο δίκτυο αισθητήρων (wireless sensor network) αποτελείται από αυτόνομους αισθητήρες οι οποίοι παρακολουθούν φυσικές ή περιβαλλοντικές συνθήκες όπως η θερμοκρασία, ο ήχος, η πίεση κ.α. και συνεργάζονται μεταξύ τους για να μεταβιβάσουν τα δεδομένα σε μία συγκεκριμένη τοποθεσία ώστε να αναλυθούν [16].

Το Διαδίκτυο των Πραγμάτων (Internet of Things – IoT) είναι ουσιαστικά μία υποκατηγορία του ασύρματου δικτύου αισθητήρων. Συγκεκριμένα με τον όρο Διαδίκτυο των Πραγμάτων αναφερόμαστε σε ένα δίκτυο από οικιακές συσκευές, οχήματα και διάφορα άλλα αντικείμενα που ενσωματώνουν λογισμικό, ηλεκτρονικά μέσα, αισθητήρες και έχουν τη δυνατότητα συνδεσιμότητας σε δίκτυο ώστε να μπορούν να ανταλλάσουν δεδομένα και να επικοινωνούν [11]. Το ευφυές δίκτυο που περιγράψαμε παραπάνω είναι πρακτικά ένα ασύρματο δίκτυο αισθητήρων που καλείται Energy Internet.

Το Διαδίκτυο των Πραγμάτων θα έχει δεσπόζουσα θέση μέσα στο πλαίσιο του ευφυούς δικτύου. Τα "έξυπνα" σπίτια του μέλλοντος θα συνεργάζονται με πολλές εξωτερικές οντότητες, όπως ανανεώσιμες πηγές ενέργειας, πάροχους ενέργειας, επιχειρήσεις κ.α.[17]. Ένα παράδειγμα για να γίνει κατανοητό πως θα συνεργάζονται τα έξυπνα σπίτια με τις διάφορες οντότητες είναι το εξής : Οι έξυπνοι μετρητές θα ενημερώνονται από τους πάροχους ενέργειας για το ποιες ώρες τις ημέρας η ηλεκτρική ενέργεια

είναι φθηνότερη. Αυτή η πληροφορία θα μεταδοθεί και στις διάφορες έξυπνες συσκευές του σπιτιού, όπως το πλυντήριο, το ηλεκτρικό όχημα κ.τ.λ. Αυτές με τη σειρά τους θα επεξεργάζονται την πληροφορία που έλαβαν και θα προγραμματίσουν ανάλογα τη λειτουργία τους .

Συνοψίζοντας τα παραπάνω μπορούμε να καταλήξουμε στο συμπέρασμα ότι το δίκτυο πάει σε μία πιο αποκεντρωμένη μορφή. Ο πολίτης πλέον θα έχει ενεργό ρόλο στην παραγωγή ενέργειας. Θα κατέχει διεσπαρμένους πόρους με τους οποίους θα μπορεί να καλύπτει τις ενεργειακές του ανάγκες αλλά και να εμπορεύεται την περισσευούμενη ενέργεια του. Τα σπίτια θα μετατραπούν σε "έξυπνα", όπου οι διάφορες συσκευές θα συνεργάζονται μεταξύ τους αλλά και με το υπόλοιπο δίκτυο. Προκύπτει λοιπόν το ζήτημα για το πώς πως θα συντονίζονται όλοι οι παράγοντες του έξυπνου δικτύου . Θα πρέπει να διασφαλιστεί ότι τα διάφορα δεδομένα τα οποία θα ρέουν εντός του δικτύου δεν αλλοιώνονται ούτε υποκλέπτονται και αποθηκεύονται με ασφάλεια. Οι πολίτες που κατέχουν διεσπαρμένους ενεργειακούς πόρους θα πρέπει να μπορούν να εμπορεύονται ενέργεια εύκολα , φθηνά και με ασφάλεια και τα προσωπικά τους δεδομένα να προστατεύονται. Από όλα τα προαναφερθέντα, προκύπτει ότι ένα σημαντικό ζήτημα, είναι το θέμα της ασφάλειας σε επίπεδο πληροφορίας, συσκευών αλλά και επικοινωνίας, έτσι ώστε να εξασφαλίζεται η ακεραιότητα και η αυθεντικότητα των δεδομένων. Αν υποθέσουμε ότι ένας κεντρικός φορέας θα αναλάβει τα παραπάνω, τότε μπορεί να προκύψουν ερωτηματικά ως προς την αξιοπιστία του ή και τις χρεώσεις των υπηρεσιών του. Έτσι λοιπόν θα πρέπει να βρεθούν αποκεντρωμένοι μέθοδοι που να διεκπεραιώνουν τις παραπάνω διαδικασίες.

# Κεφάλαιο 3

## *Τεχνολογίες Blockchain*

## 3.1 Δίκτυα υπολογιστών

Ένα δίκτυο υπολογιστών απαρτίζεται από υπολογιστές που είναι συνδεμένοι ο ένας με τον άλλον. Οι υπολογιστές του δικτύου μπορούν να χρησιμοποιούν διάφορα πρωτόκολλα για να μοιράζονται τους πόρους τους [18].

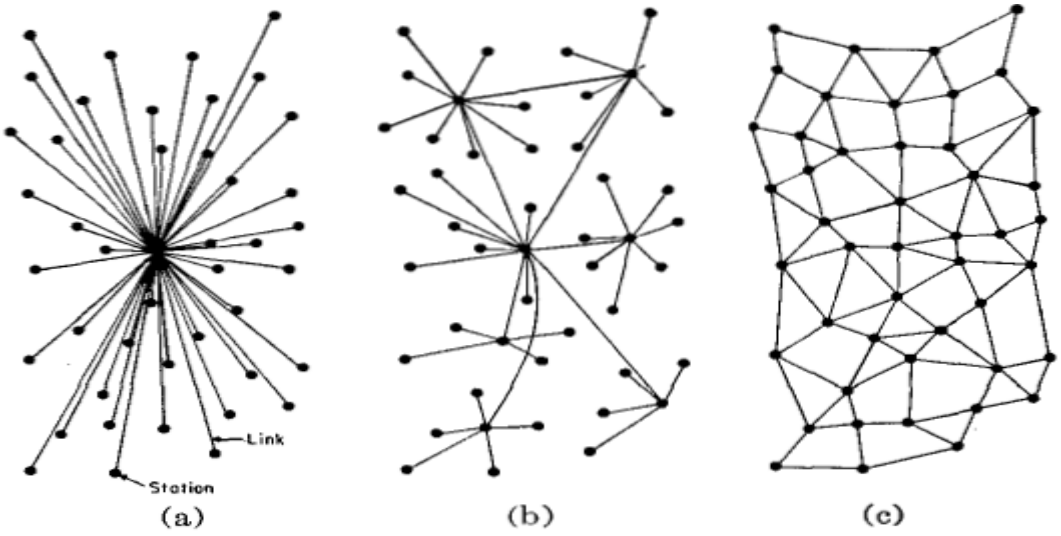


Fig. 1—(a) Centralized. (b) Decentralized. (c) Distributed networks.

Εικόνα 3.1: Κεντρικά, Αποκεντρωμένα και Κατανεμημένα Δίκτυα [18]

Μέχρι σήμερα το πιο συνηθισμένο μοντέλο με το οποίο δουλεύουν τα δίκτυα υπολογιστών είναι το μοντέλο πελάτη-εξυπηρετητή (client-server) ή αλλιώς κεντρικό μοντέλο (centralized). Στο μοντέλο αυτό υπάρχουν κόμβοι-πελάτες οι οποίοι στέλνουν αιτήματα στον εξυπηρετητή για να τους παρέχει κάποιες υπηρεσίες-πόρους και ο εξυπηρετητής ανταποκρίνεται ανάλογα. Αυτό το μοντέλο έχει το αρνητικό του single point of failure. Αν ο εξυπηρετητής αφαιρεθεί, τότε το δίκτυο καταρρέει [19].

Από την άλλη υπάρχουν και τα δίκτυα που στηρίζονται σε αποκεντρωμένο πρωτόκολλο (decentralized). Στην περίπτωση αυτή υπάρχουν περισσότεροι από ένας εξυπηρετητές και έτσι αν αφαιρεθεί ένας δεν επηρεάζεται το δίκτυο. Σε αυτό το πρωτόκολλο βασίζεται και το Διαδίκτυο [18].

Τέλος υπάρχουν και τα δίκτυα που λειτουργούν με κατανεμημένο (distributed) πρωτόκολλο. Στα εν λόγω δίκτυα ο κάθε κόμβος είναι και πελάτης και εξυπηρετητής και οι κόμβοι μοιράζονται όλους τους πόρους. Έτσι λειτουργούν και τα δίκτυα ομότιμων κόμβων (Peer to Peer –P2P) [19].

## 3.2 Εισαγωγή στο Blockchain

Το blockchain είναι ένα κατανεμημένο μητρώο (distributed ledger) από συναλλαγές (transactions), το οποίο αποθηκεύεται ταυτόχρονα από όλους τους χρήστες του [20]. Είναι ουσιαστικά μια αποκεντρωμένη ψηφιακή βάση δεδομένων η οποία ανανεώνεται από τις συναλλαγές οι οποίες συμβαίνουν μεταξύ των διαφόρων χρηστών (αφού πρώτα επικυρωθεί η ορθότητά τους) και αποθηκεύεται σε όλους τους υπολογιστές οι οποίοι συμμετέχουν στο δίκτυο και όχι σε έναν κεντρικό υπολογιστή. Η κύρια εφαρμογή στην οποία έχει χρησιμοποιηθεί μέχρι στιγμής σαν βασική τεχνολογία είναι το ψηφιακό κρυπτονόμισμα (digital cryptocurrency), bitcoin. Το bitcoin προτάθηκε το 2008 από τον Satoshi Nakamoto σαν ένα peer to peer (P2P) σύστημα πληρωμών με ψηφιακό συνάλλαγμα ανοιχτού κώδικα, το οποίο βασίζεται σε τεχνικές κρυπτογραφίας [21]. Στόχος του bitcoin είναι να καταργήσει τους ενδιάμεσους χρηματοπιστωτικούς οργανισμούς, οι οποίοι εξυπηρετούν σαν αξιόπιστα τρίτα μέρη που διαχειρίζονται τις ηλεκτρονικές πληρωμές. Όπως εξηγεί ο Nakamoto στο [22], ενώ το σύστημα λειτουργεί αρκετά καλά για τις περισσότερες συναλλαγές, πάσχει από τις αδυναμίες αυτού του μοντέλου που βασίζεται στην εμπιστοσύνη. Κάποιες από αυτές τις αδυναμίες είναι :

- Το κόστος διαμεσολάβησης που αυξάνει το κόστος της συναλλαγής, περιορίζοντας το ελάχιστο πρακτικό μέγεθος συναλλαγής και ελαχιστοποιεί την πιθανότητα για μικρές καθημερινές συναλλαγές
- Οι έμποροι θα πρέπει να είναι πολύ προσεκτικοί με τους πελάτες τους και να λαμβάνουν πολλές πληροφορίες από αυτούς .
- Συγκεκριμένο ποσοστό απάτης είναι αποδεκτό ως αναπόφευκτο.
- Οι μη αναστρέψιμες συναλλαγές είναι πρακτικά αδύνατες, επειδή οι χρηματοπιστωτικοί οργανισμοί δεν μπορούν να αποφύγουν τη διαμεσολαβητική αντιπαράθεση. Η πιθανότητα για αντιστροφή αυξάνει την ανάγκη για εμπιστοσύνη.

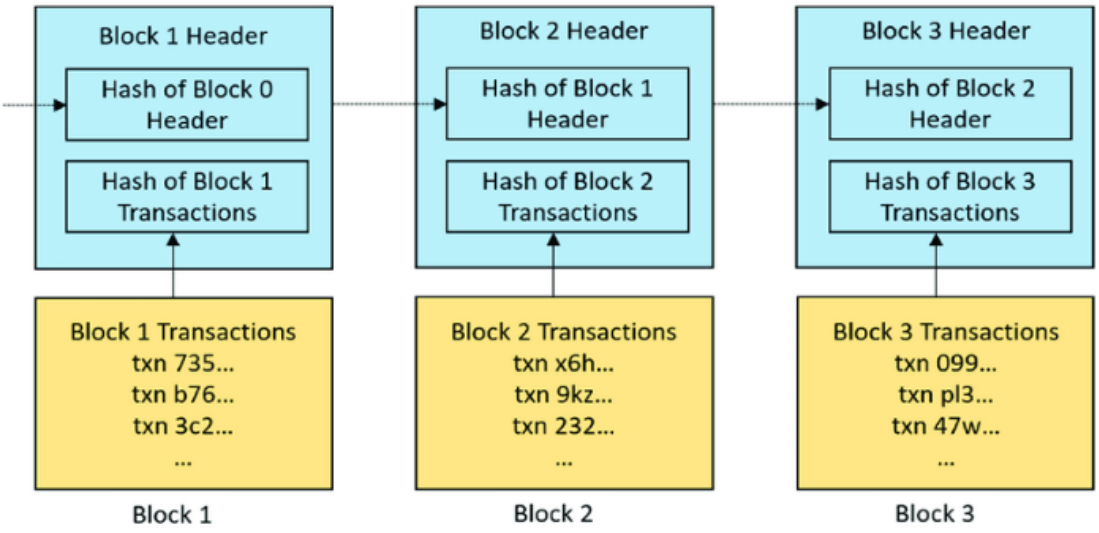
Για την εξάλειψη των παραπάνω προβλημάτων ο Nakamoto πρότεινε το σύστημα του bitcoin το οποίο βασιζόμενο στην κρυπτογραφία δεν θα απαιτεί εμπιστοσύνη και θα επιτρέπει στα δύο συμβαλλόμενα μέρη να συναλλάσσονται χωρίς την ανάγκη για αξιόπιστο τρίτο μέρος. Οι συναλλαγές θα αποθηκεύονται σε blocks (αλυσίδα από blocks) και από τη στιγμή που καταχωρούνται σε ένα block θα είναι αδύνατο να αντιστραφούν.

Γενικά το blockchain είναι μία αλυσίδα διαφορετικών blocks που αποτελείται από : τα δεδομένα του, το hash του και το hash του προηγούμενου block [23].

Τα δεδομένα που αποθηκεύονται σε κάθε block εξαρτώνται από το είδος της εφαρμογής στην οποία χρησιμοποιείται το blockchain. Για παράδειγμα στο bitcoin κάθε block αποθηκεύει οικονομικές συναλλαγές. Το μέγεθος κάθε block επίσης εξαρτάται από το είδος της εφαρμογής. Στο bitcoin κάθε block είναι ένα 1 MB.

Όσον αφορά το hash του block (θα εξηγηθεί παρακάτω τι ακριβώς είναι το hash), είναι σαν το αποτύπωμα του block και είναι χαρακτηριστικό για κάθε block. Το hash του block υπολογίζεται τη στιγμή που δημιουργείται το block. Αν αλλάξει κάτι μέσα στο block θα αλλάξει και το hash του.

Το τρίτο στοιχείο που έχει ένα block είναι το hash του προηγούμενο block (parent block). Μόνο το πρώτο block δεν έχει προηγούμενο hash και ονομάζεται block γένεσης (genesis block). Το τρίτο στοιχείο είναι που καθιστά το blockchain ως αλυσίδα από blocks. Αν αλλάξει κάτι σε ένα block τότε το hash του όπως είπαμε θα αλλάξει με αποτέλεσμα το επόμενο block να έχει διαφορετικό hash προηγούμενου block. Επομένως για να μεταβληθούν τα περιεχόμενα ενός block, πρέπει να αλλάξουν τα hash όλων των επόμενων blocks που ακολουθούν στην αλυσίδα [24].




Εικόνα 3.2: Απλοποιημένη αναπαράσταση της δομής ενός Blockchain (txn=transaction) [25]

Βέβαια το ότι κάθε block "δείχνει" στο προηγούμενο του δεν είναι αρκετό ώστε να εγγυηθεί ότι η αλυσίδα παραμένει αναλλοίωτη. Ειδικά στις μέρες μας που οι υπολογιστές είναι ταχύτατοι, θα μπορούσε κάποιος να αλλοιώσει τα δεδομένα ενός block και παράλληλα να αλλάξει τα hash όλων των επόμενων blocks έτσι ώστε να ταιριάζουν. Έτσι λοιπόν το blockchain έχει και άλλα χαρακτηριστικά που εγγυώνται την ασφάλεια και την ακεραιότητα των δεδομένων. Αυτά θα αναλυθούν στην επόμενη ενότητα.

## 3.3 Χαρακτηριστικά της τεχνολογίας του Blockchain

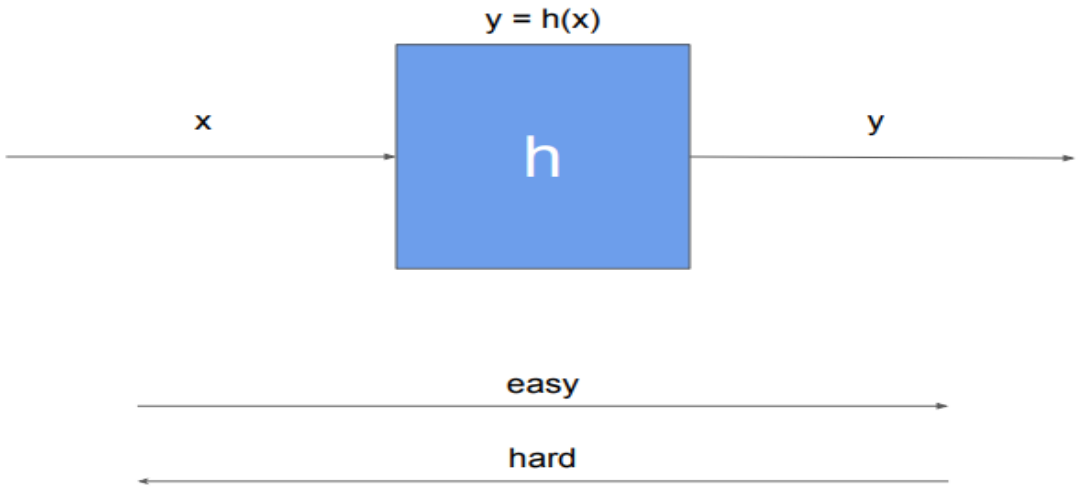
Πριν αναλύσουμε τα χαρακτηριστικά του blockchain κρίνεται σκόπιμο να αναφερθούμε στην έννοια της συνάρτηση κατακερματισμού (Hash Function) μιας και θα τη συναντάμε συνέχεια .

### 3.3.1 Συνάρτηση κατακερματισμού

Συνάρτηση κατατεμαχισμού ή κατακερματισμού (Hash function) είναι μία μαθηματική συνάρτηση η οποία με είσοδο κάποιο δεδομένο τυχαίου μεγέθους, επιστρέφει ως έξοδο μία συμβολοσειρά σταθερού μεγέθους αναπαράστασης [26]. Ουσιαστικά βάζουμε ως είσοδο μία λέξη – κλειδί και μας δίνει ως έξοδο την αντίστοιχη τιμή κατακερματισμού (hash value ή hash). Στην κρυπτογραφία ορίζουμε την κρυπτογραφική συνάρτηση κατατεμαχισμού (cryptographic hash function). Κάποιες βασικές ιδιότητες που έχουν οι συναρτήσεις κατακερματισμού είναι οι ακόλουθες [27]:

- Είναι υπολογιστικά αποδοτικές (computationally efficient). Αυτό πρακτικά σημαίνει ότι ένας μέσος υπολογιστής εκτελεί τις συναρτήσεις κατακερματισμού σε σύντομο χρονικό διάστημα. Πρέπει δηλαδή δεδομένης της εισόδου να υπολογίζει την έξοδο πολύ γρήγορα.
- Είναι ντετερμινιστικές (deterministic). Αυτό πρακτικά σημαίνει ότι για την ίδια είσοδο, η έξοδος της συνάρτησης κατακερματισμού είναι η ίδια.
- Είναι ανθεκτικές στη σύγκρουση (collision resistant). Αυτό σημαίνει ότι είναι πρακτικά αδύνατο δύο διαφορετικές είσοδοι να δώσουν την ίδια έξοδο. Ακόμα και μία μικρή αλλαγή στην είσοδο δίνει τελείως διαφορετικό αποτέλεσμα.
- Είναι ανθεκτικές στην προ εικόνα (pre-image resistant). Αυτό σημαίνει ότι η έξοδος της συνάρτησης κατακερματισμού δεν αποκαλύπτει καμία πληροφορία για την είσοδο της συνάρτησης. Δηλαδή δεδομένης της εξόδου είναι πρακτικά αδύνατο να βρούμε την είσοδο .

Στην εικόνα που ακολουθεί συνοψίζονται οι βασικές ιδιότητες της κρυπτογραφικής συνάρτησης κατακερματισμού :




Εικόνα 3.3: Συνάρτηση κατακερματισμού[28]

### 3.3.2 Ψηφιακή υπογραφή

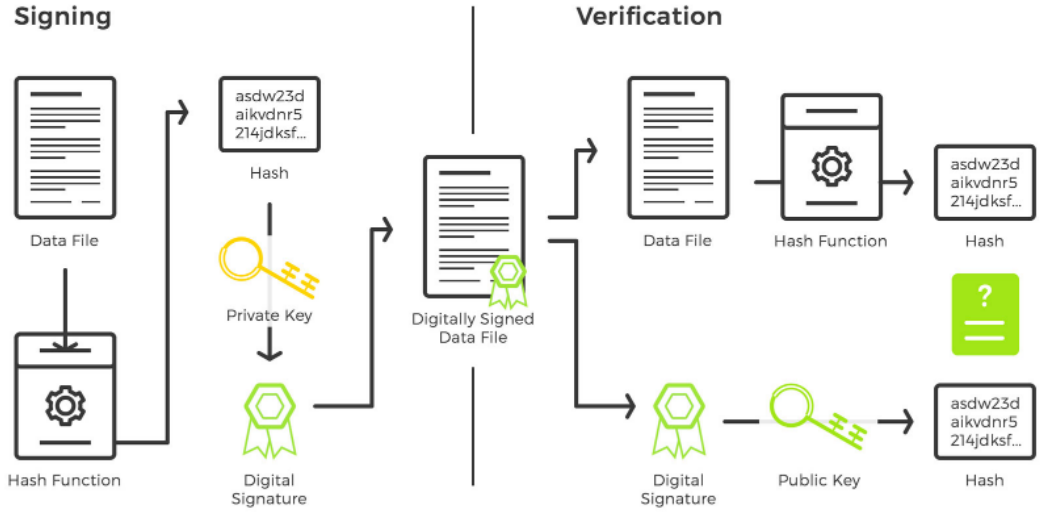
Αφού αναλύσαμε τι είναι συνάρτηση κατακερματισμού, θα δούμε πως δημιουργείται μια συναλλαγή. Καταρχάς πρέπει να αναφέρουμε ότι το blockchain χρησιμοποιεί κρυπτογραφία σαν μέσο για να διασφαλίσει ότι οι συναλλαγές γίνονται με ασφάλεια και εντιμότητα [29]. Συγκεκριμένα χρησιμοποιεί τη κρυπτογραφία ελλειπτικών καμπυλών (elliptic curve cryptography) που είναι προσέγγιση της κρυπτογραφίας δημόσιου κλειδιού (public key cryptography) που βασίζεται στην αλγεβρική δομή των ελλειπτικών καμπύλων σε πεπερασμένα πεδία [30]. Στη περίπτωση του blockchain η κρυπτογραφία ελλειπτικών καμπυλών χρησιμοποιείται μαζί με την ψηφιακή υπογραφή (digital signature) και ο αλγόριθμος ονομάζεται elliptic curve digital signature algorithm (ECDSA) [31].

Κάθε χρήστης κατέχει ένα ζευγάρι κλειδιών, το δημόσιο κλειδί (public key) και το ιδιωτικό κλειδί (private key). Το ιδιωτικό κλειδί θα πρέπει ο χρήστης να το προφυλάσσει και να το κρατάει κρυφό, ενώ το δημόσιο κλειδί μπορεί να το δώσει σε όποιον επιθυμεί [32].

Η διαδικασία λοιπόν για να γίνει μία συναλλαγή περιλαμβάνει δύο φάσεις [32] :

- τη φάση υπογραφής (signing phase)
- τη φάση επικύρωσης (verification phase)

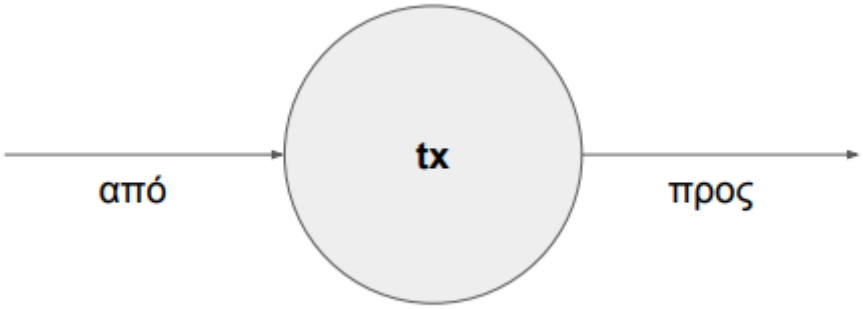
Ο χρήστης ο οποίος επιθυμεί να στείλει χρήματα υπογράφει ψηφιακά τη συναλλαγή με το ιδιωτικό του κλειδί (ουσιαστικά την κρυπτογραφεί) και τα στέλνει στο δημόσιο κλειδί του παραλήπτη (για την ακρίβεια στη διεύθυνση του, που αποτελεί το hash του δημοσίου κλειδιού του). Στην περίπτωση μιας οικονομικής συναλλαγής αποδεικνύει με αυτό τον τρόπο ότι κατέχει τα χρήματα που θέλει να στείλει. Στη συνέχεια ο παραλήπτης επικυρώνει τη ψηφιακή υπογραφή, χρησιμοποιώντας το δημόσιο κλειδί του αποστολέα, το οποίο αντιστοιχεί στο ιδιωτικό κλειδί με το οποίο ο αποστολέας υπέγραψε τη συναλλαγή [33].




Εικόνα 3.4: Ψηφιακή υπογραφή (φάση υπογραφής και φάση επικύρωσης) [34]

Στην εικόνα 3.4 βλέπουμε παραστατικά αυτό που εξηγήθηκε παραπάνω. Συνοπτικά η διαδικασία που ακολουθείται είναι: Αρχικά δημιουργείται το hash των δεδομένων και στη συνέχεια κρυπτογραφούνται με το ιδιωτικό κλειδί του αποστολέα (έτσι δημιουργείται η ψηφιακή υπογραφή). Τα δεδομένα μαζί με την ψηφιακή υπογραφή στέλνονται στη διεύθυνση του παραλήπτη. Κατά τη διαδικασία της επικύρωσης, τα δεδομένα της συναλλαγής ξεχωρίζονται από την ψηφιακή υπογραφή. Ο παραλήπτης αποκρυπτογραφεί την ψηφιακή υπογραφή με το δημόσιο κλειδί του αποστολέα και εξάγει το hash των δεδομένων. Στη συνέχεια δημιουργεί το hash των δεδομένων της συναλλαγής και εξετάζει αν τα δύο προκύπτοντα hash είναι ίδια. Στην περίπτωση που είναι ίδια, σημαίνει ότι ο αποστολέας κατέχει πράγματι το ιδιωτικό κλειδί που αντιστοιχεί στο δημόσιο με το οποίο έγινε η επικύρωση της συναλλαγής.

Στην περίπτωση του bitcoin η κάθε συναλλαγή (transaction-tx) έχει τουλάχιστον μία είσοδο και τουλάχιστον μία έξοδο [21].




Εικόνα 3.5: Συναλλαγή στο bitcoin[35]

Κάθε είσοδος μίας συναλλαγής αναγράφει ένα ποσό χρημάτων που ο αποστολέας θέλει να στείλει καθώς και τη διεύθυνση η οποία κατέχει αυτά τα χρήματα. Στη συνέχεια επιλέγει πως θέλει να διαμοιράσει αυτά τα χρήματα στις εξόδους. Προφανώς θα πρέπει το άθροισμα των χρημάτων που αναγράφονται στις εισόδους να είναι ίσο ή μεγαλύτερο από το άθροισμα των χρημάτων που αναγράφονται στις εξόδους [21].

Η έξοδος της συναλλαγής περιλαμβάνει δύο πεδία: ένα πεδίο που δηλώνει πόσα χρήματα ανήκουν στην έξοδο και ένα άλλο το οποίο ουσιαστικά είναι ένα πρόγραμμα. Το πρόγραμμα αυτό καλείται pubkey script (ουσιαστικά είναι λίστα από εντολές) και ορίζει τον τρόπο με τον οποίο ο παραλήπτης θα μπορεί να ξοδέψει τα χρήματα αυτά αργότερα. Συνήθως το script συσχετίζει τα bitcoins με μια διεύθυνση και επιτρέπει στον κάτοχο του αντίστοιχου ιδιωτικού κλειδιού να τα ξοδέψει. Γενικά οι ακμές εξόδου μια συναλλαγής μπορούν να χρησιμοποιηθούν ως είσοδοι σε περαιτέρω συναλλαγές. Οι έξοδοι που δεν έχουν χρησιμοποιηθεί καλούνται αξόδευτες έξοδοι συναλλαγών (Unspent Transaction Outputs - UTXOs) [21].

Συνοψίζοντας ένας χρήστης για να κάνει μια συναλλαγή θα πρέπει να ακολουθήσει τα ακόλουθα βήματα:

1) να βρει αξόδευτες εξόδους συναλλαγών για τις οποίες κατέχει το ιδιωτικό κλειδί
2) Να εισάγει κάποιες παραμέτρους (scriptSig) στο scriptPubKey. Αυτές είναι το δημόσιο κλειδί το όποιο αν περάσει από συνάρτηση hash βγάζει την διεύθυνση στην όποια ανήκουν τα χρήματα (διεύθυνση που είχε οριστεί από τον προηγούμενο αποστολέα που έστειλε τα χρήματα στον τωρινό κάτοχο) και μία ψηφιακή υπογραφή για να πιστοποιηθεί η κατοχή του δημόσιο κλειδιού που μόλις ορίστηκε. Αν το αποτέλεσμα είναι 1 τότε μπορεί να ξοδέψει τα χρήματα [36].

Από τα παραπάνω συμπεραίνουμε ότι κάθε χρήστης για να συναλλάσσεται στο blockchain δεν χρειάζεται να κάνει γνωστό το όνομα του ή άλλα προσωπικά του στοιχεία. Αυτό που χρειάζεται είναι μια διεύθυνση bitcoin στην οποία θα λαμβάνει / ξοδεύει χρήματα αλλά και να προσέχει το αντίστοιχο ιδιωτικό κλειδί. Συνήθως αυτό φυλάσσεται μέσα στο πορτοφόλι κρυπτονομισμάτων, που είναι ένα λογισμό το οποίο αλληλεπιδρά με διαφορετικά blockchain και επιτρέπει στο χρήστη να κάνει τις συναλλαγές του και καταγράφει το υπόλοιπό του. Επίσης ένα πορτοφόλι μπορεί να δημιουργήσει μεγάλο αριθμό διευθύνσεων επιτρέποντας στο χρήστη να συναλλάσσεται από πολλές διευθύνσεις και άρα να διατηρεί καλύτερα την ανωνυμία του [37].

Ένας χρήστης του δικτύου αφού δημιουργήσει μία συναλλαγή θα πρέπει να την μεταδώσει στους γειτονικούς του κόμβους και αυτοί με τη σειρά τους στους δικούς τους (με την προϋπόθεση ότι αυτή είναι έγκυρη) μέχρι να μαθευτεί σε όλο το δίκτυο. Στη συνέχεια θα πρέπει να περιμένει μέχρι αυτή η συναλλαγή να καταχωρηθεί σε block. Η διαδικασία αυτή με την οποία οι συναλλαγές καταχωρούνται σε blocks λέγεται εξόρυξη (mining).

### 3.3.3 Εξόρυξη blocks

Η εξόρυξη των blocks γίνεται από τους λεγόμενους μεταλλωρύχους (miners). Πριν αναφερθούμε στους μεταλλωρύχους θα πρέπει να ορίσουμε τι είναι πλήρεις και τι ελαφροί κόμβοι (full και light nodes αντίστοιχα) [38]:

- Πλήρης κόμβος (full node) είναι κάθε κόμβος (χρήστης του δικτύου) που αποθηκεύει όλα τα δεδομένα του blockchain και μπορεί να επαληθεύει την εγκυρότητα όλων των συναλλαγών αλλά και να συμμετέχει στη διαδικασία της εξόρυξης. Οι πλήρεις κόμβοι είναι αυτοί που στηρίζουν το δίκτυο μιας και αποθηκεύουν όλο το blockchain και συνδράμουν στην περεταίρω ανάπτυξή του. Όσοι περισσότεροι πλήρεις κόμβοι υπάρχουν στο δίκτυο τόσο πιο αποκεντρωμένο γίνεται .
- Ελαφρύς κόμβος (light node) είναι κάθε κόμβος ο οποίος δεν αποθηκεύει όλο το blockchain αλλά μόνο ένα υποσύνολο των blocks και στηρίζεται στους πλήρεις κόμβους για λειτουργίες που απαιτούν ολόκληρο το blockchain. Ένας ελαφρύς κόμβος εφαρμόζει την απλοποιημένη επικύρωση πληρωμών (Simplified Payment Verification-SPV), στην οποία αποθηκεύονται μόνο οι επικεφα-

λίδες των blocks κατά τη διάρκεια του αρχικού συγχρονισμού. Με την απλοποιημένη επικύρωση συναλλαγών οι ελαφροί κόμβοι μπορούν μόνο να επικυρώσουν ότι μια συναλλαγή έχει μπει στο blockchain. Αυτό του είδους κόμβοι είναι ιδιαίτερα χρήσιμοι σε περιπτώσεις όπου δεν είναι εφικτή η αποθήκευση ενός τόσο μεγάλου αρχείου όπως είναι το blockchain, π.χ. σε ένα κινητό τηλέφωνο.

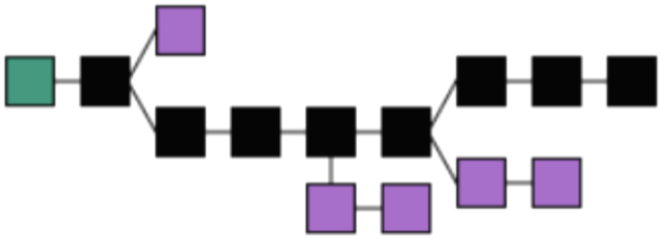
Ο μεταλλωρύχος λοιπόν είναι πλήρης κόμβος ο οποίος ομαδοποιεί τις συναλλαγές σε block και συναγωνίζεται τους υπόλοιπους μεταλλωρύχους για το ποιος θα λύσει ένα κρυπτογραφικό πάζλ γρηγορότερα, έτσι ώστε το δικό του block να προστεθεί στην αλυσίδα. Συγκεκριμένα αφού συλλέξει τις συναλλαγές τις οποίες θέλει να εντάξει σε block θα πρέπει να παράγει το χαρακτηριστικό hash αυτού του block, το οποίο θα πρέπει να είναι μικρότερο από ένα νούμερο που ονομάζεται στόχος target [39],[21]. Ουσιαστικά έχει να λύσει την παρακάτω εξίσωση [35] :

$$H(txs||\ nonce||parent\ block\ hash) < T \quad (3.1)$$

Τα hash των έγκυρων blocks στο blockchain ξεκινάει με ένα συγκεκριμένο πλήθος μηδενικών. Έτσι λοιπόν ο μεταλλωρύχος αλλάζει την παράμετρο nonce μέχρις ώστε να παράγει ένα block hash με τα απαραίτητα μηδενικά στην αρχή. Το nonce είναι μία τυχαία τιμή μεγέθους 32-bits (καταχωρείται και στην επικεφαλίδα του block) που οι μεταλλωρύχοι μεταβάλλουν ώστε να βρουν ένα block hash που ικανοποιεί τη σχέση (3.1). Όταν τελικά ένας miner βρει ένα τέτοιο block το δημοσιοποιεί στο δίκτυο και αν πράγματι είναι έγκυρο τότε οι χρήστες του δικτύου το δέχονται και ενημερώνουν την αλυσίδα τους. Τότε λέμε ότι το δίκτυο πέτυχε συναίνεση (consensus) [21].

Όταν ένας μεταλλωρύχος δημιουργήσει ένα έγκυρο block το οποίο έγινε δεκτό από το δίκτυο τότε ανταμείβεται με ένα συγκεκριμένο ποσό ως ανταμοιβή για την υπολογιστική ισχύ που ξόδεψε για να υπολογίσει το hash του block και επίσης με κάποια τέλη που πληρώνει ο κάθε χρήστης που κάνει μία συναλλαγή έτσι ώστε να "δελεάσει" το μεταλλωρύχο να συμπεριλάβει τη συναλλαγή στο block του [40]. Το ποσό που δίνεται τη παρούσα περίοδο στο bitcoin για τη δημιουργία ενός block είναι 12,5 BTC. Έτσι παράγεται χρήμα στο σύστημα του blockchain. Βέβαια το ποσό των 12,5 BTC θα μειώνεται κατά 50% ανά περίπου 4 χρόνια μέχρι να δημιουργηθεί και το τελευταίο bitcoin.

Υπάρχει και η περίπτωση να δημιουργηθούν την ίδια στιγμή δυο ή περισσότερα blocks και επομένως η βασική αλυσίδα του blockchain να διαιρεθεί σε 2 ή περισσότερα κομμάτια. Αυτό καλείται fork [18]. Το fork λύνεται όταν προστεθούν επόμενα blocks και ουσιαστικά μία από τις αλυσίδες γίνεται μεγαλύτερη. Σε αυτή την περίπτωση το δίκτυο δέχεται την μεγαλύτερη σε μήκος αλυσίδα.



Εικόνα 3.6: Διαφορετικές αλυσίδες από blocks (το πράσινο block είναι το genesis block). Η αλυσίδα που αποτελείται από τα μαύρα blocks είναι η κανονική μιας και έχει μεγαλύτερο μήκος [18].

### 3.3.4 Πρωτόκολλα συναίνεσης

Όπως αναφέρθηκε και παραπάνω οι χρήστες του δικτύου πρέπει να συμφωνήσουν μεταξύ τους πριν προσθέσουν ένα block στην αλυσίδα. Αυτό είναι ουσιαστικά το πρωτόκολλο συναίνεσης.

**Proof of Work**

Παραπάνω εξηγήσαμε ότι η αλυσίδα των blocks επεκτείνεται από τους μεταλλωρύχους, οι όποιοι προσπαθούν να βρουν μια τιμή η οποία όταν μπαίνει ως είσοδο σε μία συνάρτηση hash (π.χ. την SHA-256) παράγει έξοδο με συγκεκριμένο αριθμό μηδενικών. Έτσι παρέχουν "απόδειξη εργασίας" (Proof of Work) για την υπολογιστική ισχύ που κατανάλωσαν και ανταμείβονται με τα τέλη των συναλλαγών και ένα συγκεκριμένο ποσό από το εκάστοτε κρυπτονόμισμα. Αυτό το πρωτόκολλο συναίνεσης ονομάζεται Proof of Work (PoW) και προτάθηκε αρχικά το 1992 από τους Cynthia Dwork και Moni Naor στο [41] ως μέθοδος για την αντιμετώπιση των ανεπιθύμητων μηνυμάτων ηλεκτρονικού ταχυδρομείου. Βέβαια η ονομασία Proof of Work δεν αναφέρεται καθόλου στο [41]. Η ονομασία Proof of Work έγινε γνωστή το 1999 από τους Markus Jakobsson και Ari Juels στο [42] και τελικά προτάθηκε το 2009 από τον Satoshi Nakamoto στο [22] ως πρωτόκολλο συναίνεσης στο bitcoin.

Η δυσκολία της επίλυσης του κρυπτογραφικού πάζλ μεταβάλλεται με τέτοιο τρόπο ώστε να δημιουργείται ένα block ανά 10 λεπτά (αυτό ισχύει στο σύστημα του bitcoin). Αν στο δίκτυο αυξηθούν πολύ οι μεταλλωρύχοι και η υπολογιστική τους δύναμη και τα blocks δημιουργούνται γρηγορότερα, τότε η δυσκολία αυξάνεται [22]. Το πρωτόκολλο που βασίζεται στην απόδειξη εργασίας είναι ασφαλές, εφόσον η υπολογιστική δύναμη των έντιμων μεταλλωρύχων είναι μεγαλύτερη από το μισό της συνολικής υπολογιστικής δύναμής του δικτύου [11],[22],[39]. Προκειμένου κάποιος να επηρεάσει το δίκτυο θα πρέπει να καταλάβει το 51% της συνολικής υπολογιστικής δύναμης. Αυτή είναι η λεγόμενη 51% επίθεση (51% attack). Βέβαια αυτό είναι αρκετά δύσκολο όσο οι χρήστες του δικτύου είναι πολλοί. Έτσι λοιπόν όσο μεγαλύτερο είναι το δίκτυο τόσο πιο ανθεκτικό είναι απέναντι σε τέτοιες επιθέσεις [43]

Η απόδειξη εργασίας έχει και κάποια μειονεκτήματα πέρα από τα πλεονεκτήματα της. Αρχικά πρέπει να σημειωθεί ότι η απόδειξη εργασίας είναι μία πολύ ενεργοβόρος δι-

αδικασία. Συνολικά η απόδειξη εργασίας για τα τρία μεγαλύτερα κρυπτονομίσματα (bitcoin, ethereum, litecoin) απαιτεί περίπου 55TWh ετησίως ποσότητα ενέργειας η οποία μπορεί να καλύψει τις ενεργειακές ανάγκες 5 εκατομμύρια νοικοκυριών στις Ηνωμένες Πολιτείες [44].

Πέρα από την αυξημένη ενεργειακή κατανάλωση που έχει η διαδικασία της απόδειξης εργασίας ένα άλλο σημαντικό ζήτημα είναι οι πισίνες εξόρυξης (mining pools) που δημιουργούνται από πολλούς μεταλλωρύχους, οι οποίοι δουλεύουν μαζί με σκοπό να εξορύξουν ένα block πιο γρήγορα και μοιράζονται τα κέρδη από την εξόρυξή του [45]. Έτσι λοιπόν ενέχει ο κίνδυνος για έλεγχο του blockchain από έναν κεντρικό φορέα αν αυτές οι πισίνες εξόρυξης ενωθούν μεταξύ τους [46].

Για να αντιμετωπιστούν αυτά τα μειονεκτήματα που έχει η απόδειξη εργασίας έχουν προταθεί και άλλα πρωτόκολλα συναίνεσης, όπως το proof of stake που θα εξηγηθεί παρακάτω.

## Proof of stake

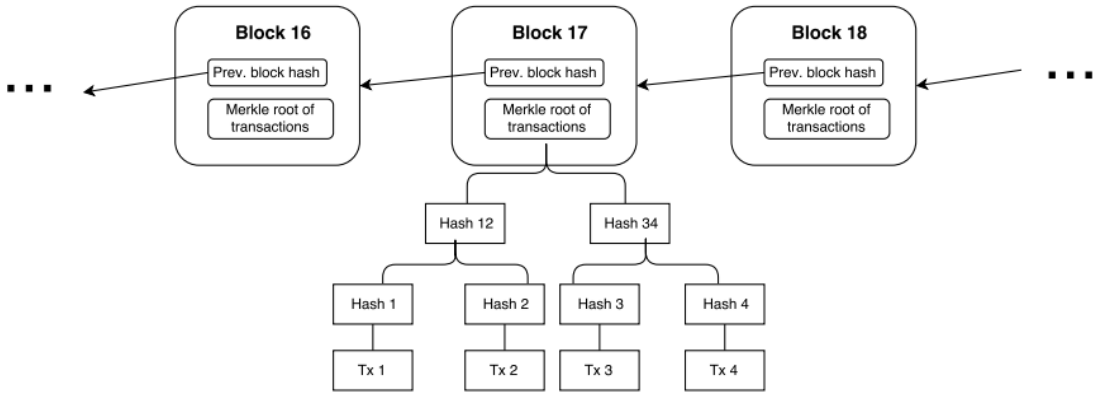
Στο πρωτόκολλο συναίνεσης Proof of Stake ένας κόμβος ο οποίος έχει δημιουργήσει ένα block θα πρέπει να αποδείξει ότι έχει πρόσβαση σε μία συγκεκριμένη ποσότητα κρυπτονομισμάτων. Οι χρήστες οι οποίες θέλουν να εξορύξουν ένα block πρέπει να στοιχηματίσουν (stake) ένα ποσό από τα κρυπτονομίσματά τους έτσι ώστε να έχουν τη δυνατότητα να επιλεχτούν για να εξορύξουν το block και να λάβουν την ανταμοιβή τους (τέλη συναλλαγών + βραβείο για την εξόρυξη του block) [47]. Έτσι λοιπόν αντί οι μεταλλωρύχοι (που στο πλαίσιο της απόδειξης στοιχήματος ονομάζονται επικυρωτές (validators)) να επενδύουν σε υπολογιστική ισχύ, στοιχηματίζουν ένα ποσό έτσι ώστε να μπορούν να επιλεχτούν για να εξορύξουν το block. Όσο μεγαλύτερο είναι αυτό το ποσό που στοιχηματίζουν τόσες περισσότερες οι πιθανότητες να επιλεχθούν για να εξορύξουν το block.

Το μεγάλο πλεονέκτημα αυτού του πρωτοκόλλου συναίνεσης είναι ότι δεν σπαταλούνται τεράστια ποσά ενέργειας στη διαδικασία της εξόρυξης και έτσι είναι φιλικό προς το περιβάλλον. Επίσης η πιθανότητα για 51% επίθεση εξαλείφεται μιας και για να γίνει αυτό θα πρέπει ο κακόβουλος επικυρωτής να κατέχει το 51% των συνολικών κρυπτονομισμάτων που υπάρχουν στο δίκτυο. Πέρα από το γεγονός ότι αυτό είναι πολύ δύσκολο έως ακατόρθωτο, επιπλέον ένας χρήστης που θα διέθετε ένα τέτοιο υπέρογκο ποσό δεν θα ήταν προς συμφέρον του να γίνει τέτοιου είδους επίθεση στο δίκτυο μιας και τότε η αξία του κρυπτονομίσματος θα "έπεφτε" [48].

Και με το πρωτόκολλο αυτό βέβαια ενέχει ο κίνδυνος να ελέγχεται το δίκτυο από ένα κεντρικό παράγοντα, που στην περίπτωση αυτή είναι οι πλουσιότεροι χρήστες [49]. Γι' αυτό οι χρήστες δεν επιλέγονται μόνο ανάλογα με το ποσό που έχουν στοιχηματίσει αλλά εκτελείται και μια τυχαία διαδικασία εκλογής. Γενικά ερευνώνται πολλές τεχνικές σχετικά με αυτή τη διαδικασία εκλογής, όπως η χρήση εντροπίας [50].

## 3.4. Το Blockchain ως δομή δεδομένων

Όπως αναφέρθηκε και προηγουμένως τα blocks αποθηκεύουν συναλλαγές. Συγκεκριμένα, αποτελούνται από δύο μέρη: την επικεφαλίδα και από όλες τις συναλλαγές οι οποίες υπάρχουν στα blocks. Η επικεφαλίδα περιέχει μεταξύ άλλων το hash του προηγούμενου block , το nonce που βρήκε ο miner και τη δυσκολία επίλυσης του αλγόριθμου Proof of Work. Επίσης αναφέραμε ότι στο δίκτυο υπάρχουν κάποιο κόμβοι (οι light nodes) οι οποίοι δεν αποθηκεύουν ολόκληρα τα blocks αλλά μόνο τις επικεφαλίδες. Προκειμένου να μπορούν να μπορούν να επαληθεύουν τις συναλλαγές χωρίς να απαιτείται να τις αποθηκεύουν όλες, στην επικεφαλίδα υπάρχει το hash της ρίζας του δέντρου Merkle (Merkle tree) των συναλλαγών του block. Δέντρο Merkle είναι ένα δέντρο όπου κάθε κόμβος - φύλλο φέρει ως ετικέτα το hash των δεδομένων που περιέχει ενώ κάθε κόμβος που δεν είναι φύλλο φέρει ως ετικέτα το hash των ετικετών των παιδιών του.




Εικόνα 3.7: Οι συναλλαγές ενός block αποθηκευμένες σε δέντρο Merkle [21]

Ουσιαστικά με αυτό τον τρόπο ένας ελαφρύς κόμβος μπορεί να επικυρώσει την εγκυρότητα μιας συναλλαγής ζητώντας τη διαδρομή που οδηγεί από τη συναλλαγή στη ρίζα [21]. Για παράδειγμα αν θέλει να επικυρώσει την εγκυρότητα της συναλλαγής 2 (Tx 2) θα ζητήσει το Hash1και το Hash 34 και έτσι θα μπορεί να επαληθεύσει τη συναλλαγή 2 χωρίς να έχει αποθηκεύσει όλο το δέντρο Merkle .

## 3.5 Κατηγορίες Blockchain

Σύμφωνα με το [51] , τα blockchains χωρίζονται σε τέσσερις κατηγορίες:

- Τα δημόσια (public) blockchains: Σε ένα δημόσιο blockchain μπορεί ο οποιοσδήποτε να διαβάζει δεδομένα αλλά και να εκτελεί συναλλαγές.
- Τα ιδιωτικά (private) blockchains: Σε ένα ιδιωτικό blockchain μπορούν να διαβάζουν δεδομένα αλλά και να εκτελούν συναλλαγές μόνο συγκεκριμένοι χρήστες, οι οποίοι συνήθως ορίζονται από πριν σε μία λίστα.
- Τα blockchain χωρίς άδεια (permissionless): Σε ένα τέτοιο blockchain μπορεί ο καθένας να δημιουργεί blocks.

- Τα blockchain με άδεια (permissioned): Σε ένα τέτοιο blockchain μπορούν να δημιουργούν blocks μόνο συγκεκριμένοι χρήστες οι οποίοι συνήθως δηλώνονται από πριν σε λίστα.

## 3.6 Ethereum

### 3.6.1 Εισαγωγή στο Ethereum

Στις ενότητες που προηγήθηκαν, παρουσιάστηκαν συνοπτικά τι είναι η τεχνολογία blockchain, κάποια βασικά χαρακτηριστικά της καθώς και κάποια χαρακτηριστικά της εφαρμογής στην οποία χρησιμοποιείται κατά κόρον, το bitcoin. Το blockchain όμως έχει αρχίσει να εφαρμόζεται και σε άλλα ψηφιακά νομίσματα (altcoins), όπως το Litecoin, το Ethereum και το Zcash. Πέρα όμως από των τομέα των οικονομικών συναλλαγών, το blockchain έχει αρχίσει να εφαρμόζεται και σε άλλους τομείς, όπως η ιδιοκτησία μιας υποκείμενης φυσικής συσκευής (έξυπνη περιουσία-smart property), σε ονόματα τομέα όπως το Namecoin αλλά και σε πιο περίπλοκες εφαρμογές οι οποίες περιλαμβάνουν περιουσιακά στοιχεία που ελέγχονται από κομμάτια κώδικα τα οποία ορίζουν αυθαίρετους κανόνες (έξυπνα συμβόλαια-smart contracts) .

Το Ethereum είναι μία πλατφόρμα blockchain η οποία διαθέτει μία ενσωματωμένη Turing-Complete γλώσσα προγραμματισμού, η οποία μπορεί να χρησιμοποιηθεί για τη δημιουργία "συμβολαίων" (κώδικας) , τα οποία μπορούν χρησιμοποιηθούν για την λειτουργία όλων των παραπάνω συστημάτων αλλά και πολλών ακόμα. Έτσι λοιπόν το Ethereum έχει σκοπό να δημιουργήσει ένα εναλλακτικό πρωτόκολλο για τη δημιουργία αποκεντρωμένων εφαρμογών (Decentralized Applications-Dapps), όπου ο καθένας θα μπορεί να γράψει το δικό του κώδικα, "χτίζοντας" τη δίκια του αποκεντρωμένη εφαρμογή, η οποία θα ορίζεται από τους δικούς τους κανόνες [52].

### 3.6.2 Λογαριασμοί

Το Ethereum μπορεί να θεωρηθεί σαν ένα σύστημα μετάβασης κατάστασης (transition state system), όπου η "κατάσταση" αποτελείται από αντικείμενα που ονομάζονται λογαριασμοί (accounts). Κάθε λογαριασμούς έχει μία διεύθυνση (address) μεγέθους 20 bytes και οι μεταβιβάσεις κατάστασης είναι απευθείας μεταβιβάσεις αξίας και πληροφορίας μεταξύ των λογαριασμών [52].

Ένας λογαριασμός αποτελείται από τα ακόλουθα τέσσερα πεδία [52]:

- Το τρέχον υπόλοιπο του σε $ether$ ($ether$ balance).
- Το nonce, που είναι ένας μετρητής που διασφαλίζει ότι κάθε συναλλαγή μπορεί να διεκπεραιωθεί μόνο μια φορά (ουσιαστικά είναι ο συνολικός αριθμός των συναλλαγών που έχουν σταλεί από τον εν λόγω λογαριασμό).
- Ο κώδικας του συμβολαίου του λογαριασμού, αν υπάρχει.
- Ο αποθηκευτικός χώρος (storage) του λογαριασμού, που από προεπιλογή είναι μηδέν.

Το ether που αναφέραμε παραπάνω είναι το εσωτερικό κρυπτονόμισμα του Ethereum και χρησιμοποιείται για την πληρωμή των τελών των συναλλαγών. Υποδιαίρεση του $ether$ είναι το 1 $wei$ και η μεταξύ τους σχέση είναι :

$$1\ ether = 10^{18} wei$$

Υπάρχουν δύο είδη λογαριασμών [52]:

- Εξωτερικοί λογαριασμοί (externally owned accounts), οι οποίοι ελέγχονται από ιδιωτικά κλειδιά.
- Λογαριασμοί των συμβολαίων (contract accounts), οι οποίοι ελέγχονται από τον κώδικα τους.

Ένας εξωτερικός λογαριασμός δεν έχει κώδικα και κάποιος μπορεί να στείλει μήνυμα (message) από έναν τέτοιο λογαριασμό δημιουργώντας και υπογράφοντας μια συναλλαγή με τον τρόπο που είδαμε στην ενότητα 3.2.2. Από την άλλη σε έναν λογαριασμό συμβολαίου κάθε φορά που λαμβάνει ένα μήνυμα τότε ενεργοποιείται ο κώδικάς του, επιτρέποντας του να διαβάσει ή να γράψει στον εσωτερικό αποθηκευτικό του χώρο και να στείλει άλλα μηνύματα ή να δημιουργήσει νέα συμβόλαια [52]. Ένα συμβόλαιο δεν μπορεί να στείλει μηνύματα από μόνο του. Πρέπει πρώτα να έχει λάβει μήνυμα από άλλο συμβόλαιο ή εξωτερικό λογαριασμό ώστε να μπορεί να δημιουργήσει συναλλαγές.

### 3.6.3 Εικονική Μηχανή του Ethereum

Στο επίκεντρο του Ethereum βρίσκεται η Εικονική Μηχανή του Ethereum (Ethereum Virtual Machine). Στην Εικονική Μηχανή του Ethereum "τρέχει" και ο κώδικας των συμβολαίων αφού πρώτα μεταγλωττιστεί σε μια γλωσσά χαμηλού επιπέδου που βασίζεται σε στοίβα και ονομάζεται bytecode Εικονικής Μηχανής του Ethereum (Ethereum Virtual Machine Bytecode - EVM Bytecode) [52].

Η Εικονική Μηχανή του Ethereum είναι quasi-Turing Complete μηχανή. Είναι λοιπόν μια μηχανή η οποία μπορεί να εκτελέσει οποιοδήποτε πρόγραμμα δίνοντας του χρόνο και μνήμη αλλά ο υπολογισμός είναι δεσμευμένος μέσω μίας παραμέτρου, το gas (θα εξηγηθεί αναλυτικά παρακάτω), που περιορίζει τους συνολικούς υπολογισμούς που γίνονται [53].

Ο κώδικας λοιπόν των συμβολαίων μεταγλωττίζεται σε σειρές από bytes, όπου κάθε byte αναπαριστά μια λειτουργία και εκτελείται ταυτόχρονα σε όλους τους κόμβους του δικτύου. Η εκτέλεση κώδικα στην εικονική μηχανή γενικά είναι πολύ εύκολη. Η κατάσταση εκτέλεσης της εικονικής μηχανής του Ethereum μπορεί να χαρακτηριστεί από την ακόλουθη πλειάδα (tuple): (block_state, transaction, message, code, memory, stack, pc, gas). Το block_state είναι η καθολική κατάσταση (global state) που περιλαμβάνει όλους τους λογαριασμούς καθώς και τα υπόλοιπα τους αλλά και τον αποθηκευτικό χώρο (storage). Το στοιχείο code είναι ουσιαστικά ο κώδικας των συμβολαίων, που περιλαμβάνει σειρές από bytes με κάθε byte να αντιπροσωπεύει μια λει-

τουργία. Στην αρχή κάθε εκτέλεσης ο δείκτης των εντολών (program counter - pc) ξεκινάει από το μηδέν και αυξάνεται κατά ένα μέχρι το τέλος του προγράμματος, εκτός αν υπάρξει κάποιο λάθος ή μια εντολή STOP ή RETURN. Κάθε byte που "δείχνεται" από το δείκτη pc είναι η αντίστοιχη εντολή που εκτελείται επηρεάζοντας ανάλογα κάθε στοιχείο της πλειάδας [52].

Η εικονική μηχανή διαθέτει τρία είδη αποθήκευσης [52] :

- Τη μνήμη (memory), που είναι ένας απείρως επεκτάσιμος πίνακας από bytes
- Τον αποθηκευτικό χώρο (storage)
- Τη στοίβα (stack), που είναι last in first out (lifo).

Ο αποθηκευτικός χώρος storage είναι εκεί όπου τα έξυπνα συμβόλαια αποθηκεύουν τις μεταβλητές κατάστασης και παραμένει αμετάβλητος μεταξύ των κλήσεων των συναρτήσεων. Βέβαια κοστίζει αρκετά για να χρησιμοποιηθεί. Η μνήμη είναι πολύ φθηνότερος τρόπος αποθήκευσης αλλά διαγράφεται μεταξύ των κλήσεων συναρτήσεων. Η στοίβα είναι περιορισμένη σε μέγεθος, δεν αποθηκεύει μόνιμα τα δεδομένα και είναι ο πιο φθηνός τρόπος αποθήκευσης στο Ethereum [21]

### 3.6.4 Συναλλαγές

Ο όρος συναλλαγή (transaction) στο Ethereum αναφέρεται στο υπογεγραμμένο πακέτο δεδομένων που αποθηκεύει ένα μήνυμα που αποστέλλεται από έναν εξωτερικό λογαριασμό [52]. Μία συναλλαγή περιλαμβάνει [52]:

- Μία ψηφιακή υπογραφή
- Το ποσό των ether που μεταφέρεται από τον αποστολέα στον παραλήπτη
- Τον παραλήπτη (recipient) του μηνύματος
- Ένα προαιρετικό πεδίο δεδομένων (data field)
- Μια τιμή startgas ή gaslimit, που αναπαριστά τον μέγιστο αριθμό των υπολογιστικών βημάτων που η εκτέλεση της συναλλαγής επιτρέπεται να εκτελέσει (θα εξηγηθεί παρακάτω )
- Μια τιμή gasprice, που αναπαριστά το ποσό που ο αποστολέας είναι διατεθειμένος να πληρώσει για κάθε υπολογιστικό βήμα (θα εξηγηθεί παρακάτω) . Όσο μεγαλύτερο είναι το gasprice τόσο πιο πιθανό είναι ένας μεταλλωρύχος να τη συμπεριλάβει στο block του .

Τα τρία πρώτα χαρακτηριστικά των συναλλαγών είναι γνωστά και από το bitcoin. Το πεδίο δεδομένων όπως αναφέρθηκε, είναι κενό από προεπιλογή. Η εικονική μηχανή του Ethereum όμως έχει μια λειτουργία με την οποία ένα συμβόλαιο μπορεί να έχει πρόσβαση σε αυτά τα δεδομένα.

### 3.6.5 Μηνύματα

Τα μηνύματα (messages) είναι σαν τις συναλλαγές. Η διαφορά τους είναι ότι δημιουργούνται από συμβόλαια και όχι από εξωτερικούς λογαριασμούς. Όπως εξηγεί ο Vitalik Buterin στο [52] είναι εικονικά αντικείμενα (virtual objects) τα οποία υπάρχουν μόνο μέσα στο περιβάλλον εκτέλεσης του Ethereum.

Ένα μήνυμα περιλαμβάνει :

- Τον παραλήπτη του μηνύματος
- Το ποσό σε ether που μεταβιβάζεται με το μήνυμα
- Τον αποστολέα του μηνύματος
- Ένα προαιρετικό πεδίο δεδομένων
- Μια τιμή STARTGAS

Όπως μια συναλλαγή έτσι και το μήνυμα καταλήγει στον λογαριασμό του παραλήπτη που εκτελεί τον κώδικά του. Έτσι τα συμβόλαια μπορούν να έχουν σχέσεις με άλλα συμβόλαια, όπως ακριβώς ένας εξωτερικός λογαριασμός έχει σχέσεις με έναν άλλον [52].

### 3.6.6 Gas

Όπως αναφέρθηκε και παραπάνω η εικονική μηχανή του Ethereum έχει έναν τρόπο με τον οποίο αποτρέπει έναν κώδικα να δημιουργεί ατέρμονους βρόχους ή άλλη υπολογιστική σπατάλη κώδικα. Έτσι λοιπόν κάθε συναλλαγή θα πρέπει να ορίζει ένα όριο από τα πόσα υπολογιστικά βήματα εκτέλεσης κώδικα μπορεί να χρησιμοποιήσει. Η μονάδα με την οποία μετράμε τα υπολογιστικά βήματα είναι το gas. Συνήθως ένα υπολογιστικό βήμα κοστίζει ένα gas. Ο σκοπός του gas είναι να απαιτήσει από έναν κακόβουλο χρήστη να προπληρώσει κάθε πόρο που θα καταναλώσει (υπολογισμούς, αποθήκευση) [52].

Κάθε συναλλαγή για να διεκπεραιωθεί επιτυχώς θα πρέπει το STARTGAS να είναι μεγαλύτερο από το συνολικό gas που απαιτεί η συναλλαγή για να γίνει. Εάν μετά το τέλος της συναλλαγής υπάρχει gas που δεν καταναλώθηκε τότε τα χρήματα που αντιστοιχούν στο υπολειπόμενο gas επιστρέφονται στον αποστολέα της συναλλαγής. Αν όμως το startgas δεν επαρκεί ώστε να διεκπεραιωθεί η συναλλαγή τότε η εικονική μηχανή του Ethereum κάνει revert, δηλαδή επανέρχεται στην κατάσταση που ήταν πριν συμβεί η συναλλαγή. Τα τέλη της συναλλαγής όμως πληρώνονται στον μεταλλωρύχο.

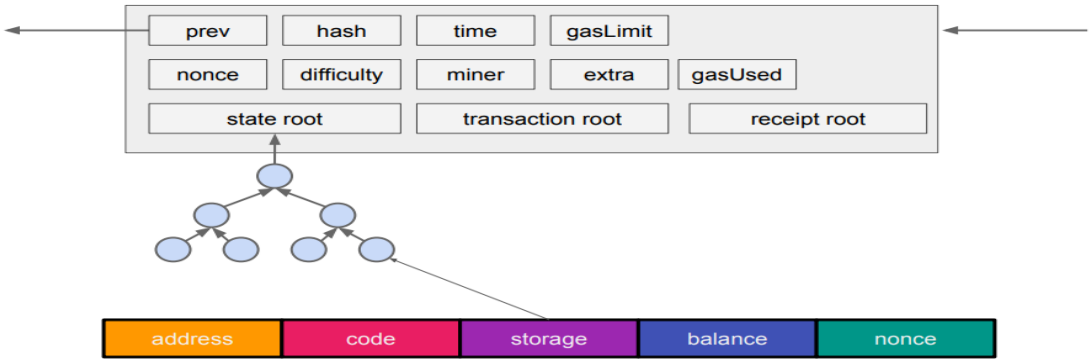
### 3.6.7 Εξόρυξη blocks

Η δημιουργία των blocks στο Ethereum είναι ανάλογη με αυτή του bitcoin. Το πρωτόκολλο συναίνεσης στο Ethereum είναι η απόδειξη εργασίας (Proof of Work). Η διαφοροποίηση σε σχέση με το bitcoin είναι ότι στο Ethereum ένα block δημιουργείται κάθε 15 με 20 δευτερόλεπτα [54]. Μια άλλη επίσης μεγάλη διαφορά είναι ότι στον

θαλγόριθμο της απόδειξης εργασίας του Ethereum, ο υπολογισμός του nonce απαιτεί πολύ μνήμη (memory- hard) και εύρος ζώνης (bandwith). Με αυτό τον τρόπο ακόμα και ένας υπερ-υπολογιστής δεν μπορεί να υπολογίζει πολλά nonce παράλληλα [55]. Αυτό δίνει τη δυνατότητα σε όλο και περισσότερους κόμβους να συμμετέχουν στην εξόρυξη blocks και άρα το δίκτυο αποκεντρώνεται ακόμα περισσότερο.

### 3.6.8 Δομή των blocks

Στην επικεφαλίδα του block στο Ethereum περά από το hash της ρίζας του Merkle δέντρου των συναλλαγών, αποθηκεύεται το hash της ρίζας του Merkle δέντρου κατάστασης (state tree) καθώς και το hash της ρίζας του Merkle δέντρου των αποδείξεων των συναλλαγών (receipt tree) [53].




Εικόνα 3.8: Δέντρο κατάστασης (State tree) [54]

Η απόδειξη μιας συναλλαγής στο Ethereum περιλαμβάνει πληροφορίες σχετικά με τη συναλλαγή . Μεταξύ άλλων περιλαμβάνει το νούμερο του block, το hash του, το hash της συναλλαγής, το gas που καταναλώθηκε για την τρέχουσα συναλλαγή κ.α.[55]

## .3.7 Εφαρμογή της τεχνολογίας Blockchain στον τομέα της ενέργειας

Στο κεφάλαιο 2 αναφερθήκαμε στην εξέλιξη των ηλεκτρικών δικτύων και στη μετάβασή τους σε μια πιο αποκεντρωμένη μορφή. Στο πλαίσιο αυτό ερευνάται η εφαρμοσιμότητα της τεχνολογίας του blockchain στα ευφυή δίκτυα. Ακολουθούν κάποιες εφαρμογές της τεχνολογία blockchain στο πλαίσιο των ευφυών δικτύων και γενικότερα στο τομέα της ενέργειας από τη διεθνή βιβλιογραφία.

### 3.7.1 Βιβλιογραφική ανασκόπηση

Στο [56] οι συγγραφείς αφού αναφερθούν στα χαρακτηριστικά του blockchain, προτείνουν τη δημιουργία ενός υβριδικού Blockchain το οποίο ονομάζεται N+X HBC (Hybrid Blockchain). Το N αντιπροσωπεύει τους ιδιωτικούς κόμβους (private nodes) και το X τους δημόσιους (public nodes). Το ιδιωτικό blockchain (εθνικοί οργανισμοί) σχεδιάζεται με σκοπό να μεταδίδει (broadcast) τις συναλλαγές και να επικυρώσει την ακρίβειά τους, ενώ το δημόσιο blockchain έχει σκοπό να επικυρώσει την ακεραιότητα των δεδομένων και τελικά να τα αποθηκεύσει.

Το [57] ασχολείται με την εφαρμογή της τεχνολογίας του blockchain στο Διαδίκτυο Ενέργειας. Αρχικά αφού οριστεί το Διαδίκτυο Ενέργειας, τονίζεται η δυνατότητα εφαρμογής του blockchain, που είναι μια αποκεντρωμένη τεχνολογία, στις αποκεντρωμένες λειτουργίες του Διαδικτύου Ενέργειας. Στη συνέχεια επισημαίνονται κάποιες ιδιότητες του Energy Internet όπως αποκέντρωση, ενσωμάτωση πολλαπλών μορφών ενέργειας στο δίκτυο, αλλά και cloud-based EMS (Energy Management System) το οποίο θα διαχειρίζεται διεργασίες επεξεργασίας δεδομένων του δικτύου και προβλήματα βελτιστοποίησης. Μετά αναφέρονται κάποια χαρακτηριστικά του blockchain (συναίνεση, ευελιξία και ασφάλεια). Στο κυρίως κομμάτι οι συγγραφείς εξηγούν το πλάνο τους για ένα πλήρως αποκεντρωμένο δίκτυο κόμβων, όπου θα υπάρχουν ηλεκτρικά οχήματα, ανεμογεννήτριες, φωτοβολταϊκά, συστήματα αποθήκευσης και απλά φορτία και το πώς θα αποθηκεύονται οι διαφορές αλλαγές στο δίκτυο π.χ. βλάβη σε γραμμή, φόρτιση ηλεκτρικών οχημάτων, σύνδεση/ αποσύνδεση κατανεμημένων διεσπαρμένων πόρων κ.α. στο blockchain και ταυτόχρονα θα αλλάζει και το μοντέλο βελτιστοποίησης στον cloud server .

Στο [58] ο συγγραφέας ασχολείται με την ανάγκη να δοθούν κίνητρα για χρήση πράσινης ενέργειας αλλά και να μειωθούν οι φραγμοί για πρόσβαση στην καθαρή ενέργεια. Καταρχάς αφού ορίσει τα Πιστοποιητικά Ανανεώσιμης Ενέργειας (RECs) και τις πιστώσεις άνθρακα (Carbon Credits), τονίζει τα εμπόδια (υψηλά κόστη και χρονοβόρος έλεγχος) που εισάγουν τα σύγχρονα συστήματα διαχείρισης και λογιστικής στην έκδοση και εξαργύρωση τους. Αυτή τη στιγμή οι πιστοποιήσεις ελέγχονται χειροκίνητα με την δημιουργία των πιστοποιήσεων αλλά και τη μεταβίβαση ιδιοκτησίας τους να καταγράφονται σε φύλλα Excel και PDF. Προτείνει τη χρήση του blockchain σαν μέσο βελτιστοποίησης των παραπάνω διαδικασιών, όπου τα δεδομένα των μετρητών θα περνάνε σε μια βάση δεδομένων, όπου θα γίνεται εξακρίβωση του μετρητή, επικύρωση της καθαρής ενέργειας και αυτόματα θα δημιουργούνται πιστοποιήσεις .

Στο [14] χρησιμοποιείται η τεχνολογία του blockchain για τη διεκπεραίωση της ανταπόκρισης ζήτησης (demand response). Συγκεκριμένα η τεχνολογία του blockchain χρησιμοποιείται για την αποθήκευση της ενεργειακής παραγωγής και κατανάλωσης, που συλλέγονται από τις έξυπνες μετρητικές συσκευές (smart meters) του Διαδικτύου των Πραγμάτων (Internet of Things ή IoT), ενώ έξυπνες συμβάσεις (smart contracts) ορίζουν την ενεργειακή ευελιξία του κάθε καταναλωτή. Κάθε φορά που η ζήτηση δεν ισούται με την παραγωγή (είτε γιατί η παραγωγή είναι μεγαλύτερη της ζήτησης είτε γιατί η παραγωγή δεν μπορεί να ανταποκριθεί στη ζήτηση ), ο διαχειριστής του συστήματος διανομής (Distribution System Operator) στέλνει στον κάθε καταναλωτή ένα σήμα (DRsignal), ώστε να μεταβάλλει την κατανάλωση του και με αυτόν τον τρόπο η ζήτηση και η παραγωγή να εξισορροπηθούν. Τα εν λόγω smart contracts ορίζουν και κάποιες επιβραβεύσεις αλλά και κάποιες ποινές στην περίπτωση που ο καταναλωτής ανταποκριθεί ή δεν ανταποκριθεί αντίστοιχα πλήρως στο σήμα που του στάλθηκε. Τέλος παρατίθενται κάποια αποσπάσματα του κώδικα αυτών των συμβολαίων που είναι γραμμένα σε Solidity, αλλά και για σκοπούς αξιολόγησης εκτελείται η προσομοίωση ενός έξυπνου δικτύου 12 κατανεμημένων ενεργειακών κα-

ταναλωτών - παραγωγών (Distributed Energy Prosumers ή DEPs) με ενεργειακά προφίλ κατανάλωσης και παραγωγής από δεδομένα Βρετανικών κτιρίων που περιγράφονται σε άλλες εργασίες. Στο πλαίσιο αυτής της αξιολόγησης καταγράφονται κάποιες καμπύλες για να φανεί η ταύτιση ζήτησης και παραγωγής όταν χρησιμοποιείται το έξυπνο συμβόλαιο που ορίζει το πως γίνεται η διαχείριση της ενέργειας από την πλευρά της ζήτησης (Demand Side Management).

Στο [20] οι συγγραφείς χρησιμοποιούν την τεχνολογία του blockchain για τη διεκπεραίωση Vicrey δημοπρασιών ενέργειας χωρίς την ανάγκη για κάποιον ενδιάμεσο (Vicrey δημοπρασία: Τύπος δημοπρασίας σφραγισμένης προσφοράς όπου οι προσφέροντες υποβάλλουν γραπτές προσφορές χωρίς να γνωρίζουν την προσφορά των άλλων προσώπων. Ο πλειοδότης κερδίζει άλλα η τιμή που ουσιαστικά πληρώνει είναι η δεύτερη υψηλότερη προσφορά. Αυτή η διαδικασία εξασφαλίζει ότι οι προσφέροντες θα κάνουν ειλικρινείς προσφορές). Αφού εξηγηθεί η διαδικασία με την οποία θα γίνεται η δημοπρασία ενέργειας, παρουσιάζεται και ένας "ψευδοκώδικας" του συμβολαίου, που θα εκτελεί τη δημοπρασία και εξηγεί τις διάφορες λειτουργίες του. Τέλος παρατίθεται και μια εφαρμογή τέτοιου τύπου δημοπρασίας που έλαβε χώρα στην πανεπιστημιούπολη της Ουάσιγκτον. Στην εν λόγω εφαρμογή, ένα φωτοβολταϊκό 72KW τοποθετημένο στο εργαστήριο ερευνών της πανεπιστημιούπολης με ενσωματωμένη συσκευή Beaglebone Black (single board computer),αλλά και άλλες δύο συσκευές Beaglebone Black εντός του εργαστηρίου θα παίξουν το ρόλο του πωλητή ενέργειας και των προσφερόντων αντίστοιχα (δυο σπίτια που θα αγοράζουν ενέργεια). Οι συσκευές έχουν εγκατεστημένη τη πλατφόρμα Volttron (πλατφόρμα που βασίζεται στο blockchain), και επικοινωνούν με το blockchain μέσω της πλατφόρμας geth (Go Ethereum). Με βάση κάποια δεδομένα που αντανακλούν τη θέληση των προσφερόντων να έχουν ένα ζεστό περιβάλλον στο σπίτι τους γίνεται μια προσομοίωση που δείχνει με βάση διάφορες προσφορές, του φωτοβολταϊκού σε KWh, πως ανταποκρίνονται τα δύο σπίτια σε αυτές (τις προφορές που υποβάλουν και τελικά την ενέργεια που αγοράζουν) .

Στο [59], οι συγγραφείς ασχολούνται με την εφαρμογή του blockchain σε μία ενεργειακή κοινότητα χαμηλής τάσης. Στόχος είναι να χρησιμοποιηθούν όσο το δυνατόν καλύτερα οι κοινοί πόροι, ώστε να μειωθεί η εξάρτηση από εξωτερική παροχή ενέργειας και όχι η εμπορεία ενέργειας. Αφού ορίσουν τι είναι ενεργειακή κοινότητα και blockchain εξηγούν την εφαρμογή τους. Στο συγκεκριμένο project, έχουμε μια κοινότητα αποτελούμενη από τέσσερα σπίτια ,όπου το υπολειπόμενο φορτίο (residual load) του κάθε σπιτιού αθροίζεται με τα άλλα ώστε να προκύψει ένα συνολικό υπολειπόμενο φορτίο της κοινότητας. Το κάθε σπίτι διαθέτει και σύστημα αποθήκευσης ενέργειας (Energy Storage System ή ESS), τα οποία λειτουργούν με αποδοτικό τρόπο, λαμβάνοντας υπόψη την φόρτιση τους. Κάθε σπίτι διαθέτει ένα φωτοβολταϊκό και ένα Raspberry Pi (single board computer), που ελέγχει το ESS και τρέχει ένα πλήρη κόμβο στο blockchain, αλλά και ένα λογισμικό διεπαφής (interface software). Ένα έξυπνο συμβόλαιο, αναπτύσσεται στο blockchain το οποίο προσφέρει εγγραφή και διαγραφή των χρηστών στο blockchain, καταγράφει την κατάσταση φόρτισης των μπατα-

ριών αλλά και το υπολειπόμενο φορτίο κάθε σπιτιού και επίσης παρέχει και τις εντολές ελέγχου που χειρίζονται τις μπαταρίες με βάση το ποιες είναι πιο φορτισμένες, έτσι ώστε το υπολειπόμενο φορτίο να καλύπτεται αποδοτικά. Τέλος εκτελούνται και κάποιες δοκιμές. Συγκεκριμένα για διάστημα μιας μέρας απεικονίζεται σε διάγραμμα το υπολειπόμενο φορτίο του κάθε σπιτιού για κάθε ώρα αυτού του διαστήματος αλλά και ποια από τα ESS κάθε φορά καλύπτουν αυτό το υπολειπόμενο φορτίο.

Στο [60] τονίζεται η ανάγκη για χρήση μια αποκεντρωμένης τεχνολογίας έτσι ώστε να εξυπηρετείται η ανταλλαγή ενέργειας μεταξύ μικρών παραγωγών και επαναφορτιζόμενων ηλεκτρικών οχημάτων. Αρχικά οι συγγραφείς κάνουν μια ανασκόπηση στο δίκτυο ηλεκτρικής ενέργειας που αλλάζει και πάει σε μια πιο αποκεντρωμένη μορφή και τονίζουν ότι το blockchain με τα smart contracts θα μπορούσαν να χρησιμοποιηθούν σαν μέσο ανταλλαγής ενέργειας μεταξύ των οντοτήτων. Προτείνουν και την εισαγωγή μια οντότητας ,την ER, η οποία δυναμικά θα προσαρμόζει τις ροές ενέργειας και θα είναι υπεύθυνη για την επικοινωνία μεταξύ των συμμετεχόντων του δικτύου. Αφού εξηγήσουν τη διαδικασία με την οποία θα γίνεται η ανταλλαγή ενέργειας αλλά και η επιβεβαίωση των συναλλαγών από τους miners, παραθέτουν δύο σενάρια. Το πρώτο σενάριο αφορά την ανταλλαγή ενέργειας μεταξύ δύο καταναλωτών σε μια συγκεκριμένη τιμή ενώ στο δεύτερο αφορά τη φόρτιση και αποφόρτιση ηλεκτρικών οχημάτων, όπου κατά την φόρτιση δύο χρήστες του δικτύου είναι συνδεδεμένοι με τον ίδιο ER ενώ κατά την αποφόρτιση οι δύο χρήστες είναι συνδεδεμένοι με διαφορετικό ER.

Στο [61] οι συγγραφείς ασχολούνται με το Διαδίκτυο των Πραγμάτων (Internet of Things ή IoT) στο πλαίσιο ενός συστήματος έξυπνου σπιτιού (Smart Home System ή SHS) και την εφαρμογή ενός ιδιωτικού blockchain, όπου θα εφαρμόζονται κάποια έξυπνα συμβόλαια τα οποία θα χειρίζονται την δυνατότητα πρόσβασης, αποθήκευσης και ροής δεδομένων μεταξύ αυτών των συσκευών. Πιο συγκεκριμένα, αφού ορίσουν τι ακριβώς είναι το IoT και το SHS, αλλά και κάποια προβλήματα ασφαλείας που υπάρχουν, αναφέρονται στην τεχνολογία του blockchain και τα βασικά χαρακτηριστικά της. Κατόπιν, γίνεται ανάλυση της προτεινόμενου συστήματος. Το συγκεκριμένο σύστημα αποτελείται από ένα SH miner (Smart Home miner),που είναι ένας υπολογιστής που διατηρεί ένα ιδιωτικό blockchain και ουσιαστικά θέτει διάφορες πολιτικές για τη ροή των δεδομένων και τη διαχείριση των συναλλαγών μεταξύ των διαφόρων συσκευών, και τοπικό αποθηκευτικό χώρο που συνδέεται με αισθητήρες και συσκευές ενεργοποίησης. Ένα έξυπνο συμβόλαιο αναπτύσσεται πάνω στο blockchain και ουσιαστικά είναι αυτό που ορίζει τις συναλλαγές δεδομένων.

Το [62] ασχολείται με το πως οι κόμβοι ενός ενεργειακού δικτύου μπορούν να ανταλλάσουν ενέργεια με βάση ένα blockchain που βασίζεται στο Tendermint. Αρχικά αναφέρονται κάποια χαρακτηριστικά του Tendermint blockchain και προτείνεται η χρήση του εντός ενός μικροδικτύου ως μητρώο ενεργειακών συναλλαγών. Συγκεκριμένα, η εφαρμογή στο κινητό του κάθε χρήστη θα επικοινωνεί με τον έξυπνο μετρητή του χρήστη, θα κάνει κάποιες προβλέψεις και θα στέλνει κάποιες προσφορές για αγορά ή πώληση ενέργειας. Οι χρήστες θα μπορούν να επιλέγουν τις προσφορές που

τους ταιριάζουν καλύτερα και έπειτα ο μετρητής θα στέλνει αίτημα συναλλαγής στο έξυπνο συμβόλαιο και αυτό θα επεξεργάζεται το αίτημα και θα στέλνει μήνυμα πίσω στο μετρητή αναφορικά με το αν το αίτημα έγινε δεκτό ή όχι.

# Κεφάλαιο 4

## *Οικονομική Κατανομή Φορτίου*

## 4.1 Εισαγωγή στην Οικονομική Κατανομή Φορτίου

Βασικός στόχος στα συστήματα ηλεκτρικής ενέργειας είναι η κάλυψη της ζήτησης φορτίου. Έτσι εντάσσονται μονάδες παραγωγής για να καλύψουν αυτή τη ζήτηση. Όμως η ένταξη μονάδων θα πρέπει να γίνεται και με τον πλέον οικονομικό τρόπο. Το κριτήριο βέλτιστης λειτουργίας είναι οικονομικό. Προσπαθούμε λοιπόν να ελαχιστοποιήσουμε το κόστος παραγωγής, χρησιμοποιώντας το καταλληλότερο καύσιμο και να μειώσουμε όσο το δυνατόν περισσότερο το κόστος μεταφοράς ελαχιστοποιώντας τις απώλειες του δικτύου [63].

Η βέλτιστη λειτουργία του συστήματος συνιστά το πρόβλημα της Οικονομικής Κατανομής Φορτίου (Economic Dispatch Problem – EDP) και βασίζεται στα ακόλουθα :

- Ακριβή πρόβλεψη φορτίων σε βραχυπρόθεσμο αλλά και μακροπρόθεσμο επίπεδο.
- Οικονομικό προγραμματισμό γεννητριών (προγραμματισμός με τον οποίο εντάσσονται οι μονάδες παραγωγής στο σύστημα).
- Οικονομική φόρτιση μονάδων που βρίσκονται σε λειτουργία (Οικονομική λειτουργία του συστήματος).

Τα τρία παραπάνω στάδια είναι άρρηκτα συνδεδεμένα. Ας αναλύσουμε λοιπόν το κάθε στάδιο ξεχωριστά.

Καταρχάς η **πρόβλεψη φορτίου** είναι πολύ σημαντική για τα συστήματα ηλεκτρικής ενέργειας έτσι ώστε να μπορούν να πραγματοποιηθούν διαδικασίες όπως η λειτουργία του συστήματος, ο ετήσιος προγραμματισμός αλλά και μελέτες επέκτασης του δικτύου. Όταν λέμε βραχυπρόθεσμη πρόβλεψη φορτίου αναφερόμαστε στη πρόβλεψη του φορτίου του συστήματος ώρα-ώρα, για κάθε συγκεκριμένη μέρα. Στη μακροπρόθεσμη πρόβλεψη φορτίου συνυπολογίζονται οι πιθανότερες ετήσιες ή εποχιακές αιχμές φορτίου σε μακροχρόνια κλίμακα. Έτσι λοιπόν η πρόβλεψη φορτίου μας επιτρέπει να γνωρίζουμε τη ζήτηση φορτίου και επομένως να κάνουμε σωστό προγραμματισμό ένταξης μονάδων. Καθίσταται λοιπόν αναγκαίο η πρόβλεψη φορτίων να γίνεται με μεγάλη ακρίβεια, ώστε να μπορούμε να επιτύχουμε την βέλτιστη λειτουργία του συστήματος.

Με τον όρο **οικονομικός προγραμματισμός γεννητριών** αναφερόμαστε στην ένταξη των μονάδων παραγωγής με τέτοιο τρόπο ώστε να εξυπηρετείται η ζήτηση του φορτίου κάθε ώρα με το ελάχιστο κόστος. Η ελαχιστοποίηση του κόστους επιτυγχάνεται με χρησιμοποίηση των λιγότερο ακριβών γεννητριών, αλλά θα πρέπει να ικανοποιείται το φορτίο και να σεβόμαστε τους περιορισμούς του συστήματος. Στη εν λόγω διαδικασία λαμβάνονται υπόψη οι περιορισμοί φορτίου, η απαραίτητη εφεδρεία αλλά επίσης το τεχνικό μέγιστο και ελάχιστο της κάθε μονάδας καθώς και οι χρόνοι ένταξης και αποσύνδεσής τους.

Τέλος έχουμε την **οικονομική κατανομή φορτίου** που αποτελεί και το αντικείμενο της συγκεκριμένης διπλωματικής. Προσπαθούμε να ρυθμίσουμε κατάλληλα τις μονά-

δες που είναι σε λειτουργία κάθε στιγμή, ώστε να καλύπτουν την απαιτούμενη ζήτηση φορτίου με τον πλέον οικονομικό τρόπο. Αυτό μεταφράζεται σε ελαχιστοποίηση του συνολικού κόστους καυσίμου, που αποτελεί και το μεγαλύτερο έξοδο ώστε να παραχθεί ηλεκτρική ενέργεια [63].

## 4.2 Κλασσικό μοντέλο Οικονομικής Κατανομής Φορτίου

Όπως έχει ήδη αναφερθεί και παραπάνω, στόχος της Οικονομικής Κατανομής Φορτίου είναι να ελαχιστοποιηθεί το λειτουργικό κόστος του συστήματος ή ισοδύναμα το κόστος κατανάλωσης καυσίμου ικανοποιώντας το φορτίο υπό τους περιορισμούς του συστήματος με δεδομένες μονάδες παραγωγής σε λειτουργία. Είναι λοιπόν ζητούμενο να αξιοποιείται κάθε φορά η οικονομικότερη μονάδα για την κάλυψη του φορτίου. Αν θεωρήσουμε λοιπόν μία απότομη αύξηση του φορτίου οι μονάδες θα πρέπει να προσαρμόσουν την παραγωγή τους με τον πιο αποτελεσματικό τρόπο έτσι ώστε η συνολική παραγωγή να καλύπτει τη ζήτηση αλλά με το ελάχιστο δυνατό κόστος. Άρα λοιπόν αυτή η απότομη αύξηση του φορτίου θα πρέπει να καλυφθεί από την πιο οικονομική μονάδα μέχρι να φτάσει τα όρια της. Στη συνέχεια η αμέσως οικονομικότερη μονάδα θα πρέπει να πρέπει να καλύψει την αύξηση του φορτίου κ.ο.κ. [64]. Έχουν αναπτυχθεί πολλοί αλγόριθμοι για να επιλύσουν το συγκεκριμένο πρόβλημα, κάποιοι από τους οποίους περιέχουν επαναληπτικές διαδικασίες, δυναμικό προγραμματισμό, εξελικτικούς αλγόριθμους και διάφορες άλλες τεχνικές. Στο παρόν κεφάλαιο θα παρουσιαστούν κάποιοι από αυτούς τους αλγορίθμους.

Κριτήριο για τη διαμοιρασμό του φορτίου στις μονάδες είναι το κατά πόσο μία αύξηση του φορτίου σε μια μονάδα και ισόποση μείωση του φορτίου σε άλλη θα επιφέρει αύξηση ή μείωση του συνολικού κόστους. Δηλαδή, αυτό που θα εξετάσουμε είναι το οριακό κόστος της κάθε μονάδας, δηλαδή θα μελετήσουμε την καμπύλη κόστους καυσίμου – παραγωγής ισχύος της κάθε μονάδας. Το κόστος της παραγωγής περιέχει μέσα και το κόστος εργασίας, της συντήρησης, της μεταφοράς του καυσίμου κτλ. Αυτά συνήθως συμπεριλαμβάνονται στον σταθερό όρο του κόστους λειτουργίας [64].

Η ενεργός ισχύς εξόδου κάθε μονάδας παραγωγής (γεννήτριας) έχει μια μέγιστη και μία ελάχιστη τιμή που καλούνται αντίστοιχα τεχνικό μέγιστο και τεχνικό ελάχιστο της γεννήτριας και θα πρέπει να ισχύει :

$$P_{i,min} \leq P_i \leq P_{i,max} \quad (4.1)$$

Για τις θερμικές μονάδες το $P_{i,min}$ προκαλείται από θερμικούς περιορισμούς στη λειτουργία του ατμοπαραγωγού και όταν αυτές εντάσσονται στο δίκτυο θα πρέπει η ισχύς που παράγουν να διατηρείται πάνω από αυτή τη τιμή. Το $P_{i,max}$ σχετίζεται περισσότερο με τη μέγιστη ισχύ που μπορεί να παράγει ο στρόβιλος της γεννήτριας και έτσι λοιπόν η παραγόμενη ισχύς θα πρέπει να διατηρείται κάτω από αυτή τη τιμή [65].

Γενικά η συνάρτηση κόστους των μονάδων είναι μη γραμμική. Εξαρτάται από τον τύπο του καυσίμου που καταναλώνεται (φυσικό αέριο, πετρέλαιο κλπ.), από τις εθνι-

κές και διεθνείς συνθήκες οικονομίας, την πολιτική κατάσταση κ.α. Η συνάρτηση κόστους της κάθε μονάδας $i$ μοντελοποιείται ως [64] :

$$C_i = a_i \cdot P_i^2 + b_i \cdot P_i + c_i + \left| e_i \cdot \sin\left(f_i(P_{i,min} - P_i)\right) \right|$$

Η παραπάνω σχέση περιλαμβάνει τους συντελεστές $e_i$ και $f_i$ που εκφράζουν την μη ομαλότητα της καμπύλης κόστους. Η σχέση αυτή δεν είναι ιδιαίτερα εύκολη για υπολογισμούς. Συνήθως για τον προσδιορισμό του κόστους χρησιμοποιείται η τετραγωνική συνάρτηση κόστους που είναι η ακόλουθη [65] :

$$C_i = a_i \cdot P_i^2 + b_i \cdot P_i + c_i \quad (4.2)$$

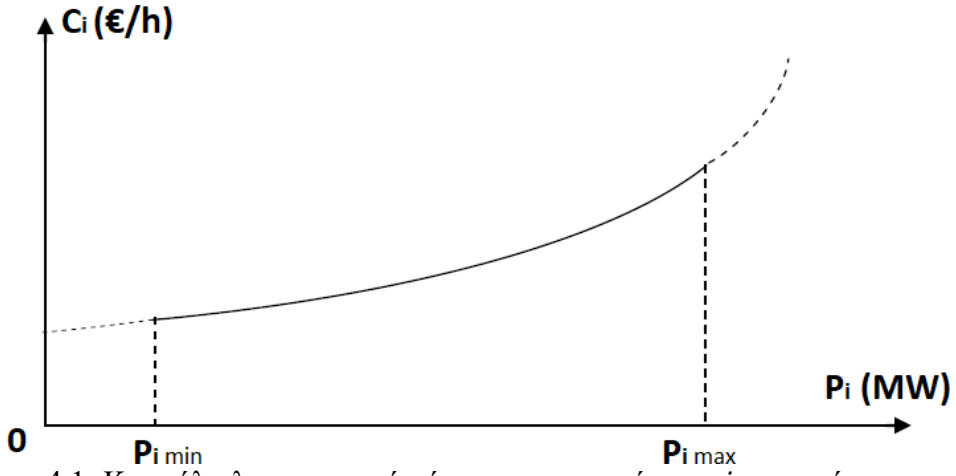
Όπου :

$C_i$: το λειτουργικό κόστος της κάθε μονάδας $i$ $€/h$.

$a_i, b_i, c_i$: οι συντελεστές της χαρακτηριστικής κόστους της κάθε μονάδας $i$ σε $€/(MW^2 \cdot h)$, $€/MWh$ και $€/h$ αντίστοιχα. Το $c_i$ αντιστοιχεί στο λειτουργικό κόστος της μονάδας για μηδενική παραγωγή.

$P_i$: η ενεργός ισχύς εξόδου της κάθε μονάδας $i$ σε $MW$.

Ακολουθεί γραφική παράσταση του κόστους όπως εκφράζεται από τη σχέση (4.2):



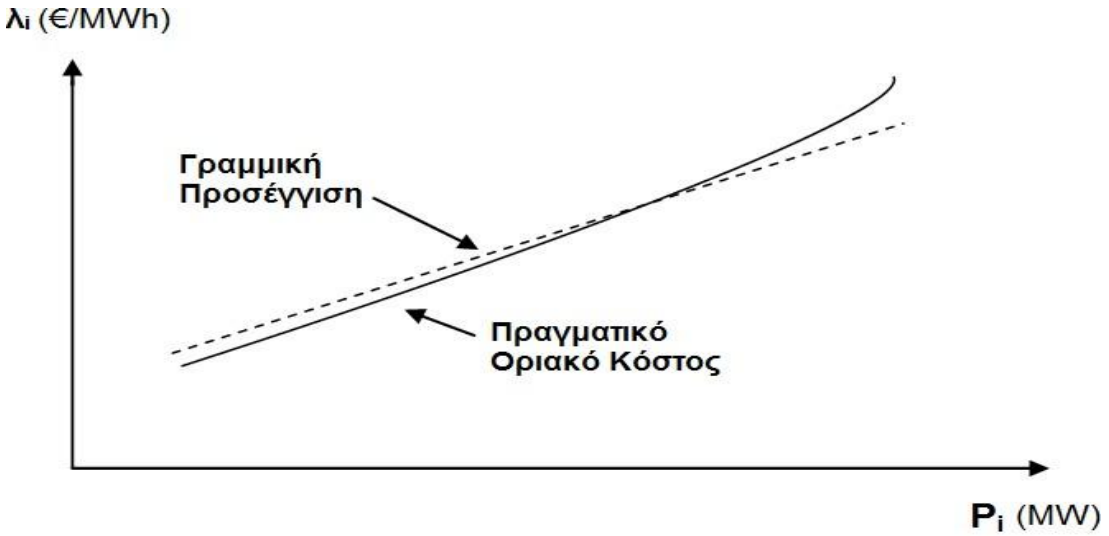

Διάγραμμα 4.1: Καμπύλη λειτουργικού κόστους της γεννήτριας $i$ σε συνάρτηση με την ενεργό παραγωγής της [66]

Με βάση την παραπάνω σχέση μπορούμε να ορίσουμε το οριακό κόστος της μονάδας $i$, που είναι η κλίση της παραπάνω καμπύλης :

$$\lambda_i = \frac{dC_i}{dP_i} = 2 \cdot a_i \cdot P_i + b_i \quad (4.3)$$

Το οριακό κόστος της μονάδας $i$ έχει μονάδες $€/MWh$ και εκφράζει το πόσο θα αυξηθεί το κόστος ανά ώρα, αν αυξηθεί το φορτίο κατά 1 $MW$. Στο διάγραμμα 4.2 βλέ-

πουμε το πως μεταβάλλεται το οριακό κόστος σε συνάρτηση με την ενεργό παραγωγή της γεννήτριας $i$.




Διάγραμμα 4.2: Οριακό κόστος λειτουργίας της γεννήτριας $i$ συναρτήσει της ενεργού παραγωγής της [64]

Από το διάγραμμα 4.2 παρατηρούμε ότι το οριακό κόστος είναι γραμμικό στο μεγαλύτερο εύρος της ισχύος εξόδου. Βέβαια στην πράξη χρησιμοποιείται κυρίως η γραμμική προσέγγιση του, η οποία δεν διαφέρει και ιδιαίτερα από το πραγματικό οριακό κόστος.

Θεωρούμε ότι το σύστημά μας αποτελείται από $N$ μονάδες παραγωγής, οπού η κάθε μία έχει παραγόμενη ισχύ $P_i$ και συνάρτηση κόστους $C_i$. Έτσι λοιπόν το συνολικό λειτουργικό κόστος του συστήματος είναι :

$$C_{total} = \sum_{\iota=1}^{N} C_i \quad (4.4)$$

Επίσης η συνολική παραγόμενη ισχύς του συστήματος είναι :

$$P_{total=} \sum_{\iota=1}^{N} P_i \quad (4.5)$$

Η συνολική παραγόμενη ισχύς θα πρέπει να καλύπτει όχι μόνο τη ζήτηση φορτίου αλλά και να αντισταθμίζει τις απώλειες του συστήματος μεταφοράς (στο πλαίσιο όμως της παρούσας διπλωματικής οι απώλειες αγνοούνται). Έτσι λοιπόν θα πρέπει να ισχύει :

$$P_{total} = P_D + P_{loss} \quad (4.6)$$

Όπου:

$P_D$: η ζήτηση φορτίου του συστήματος

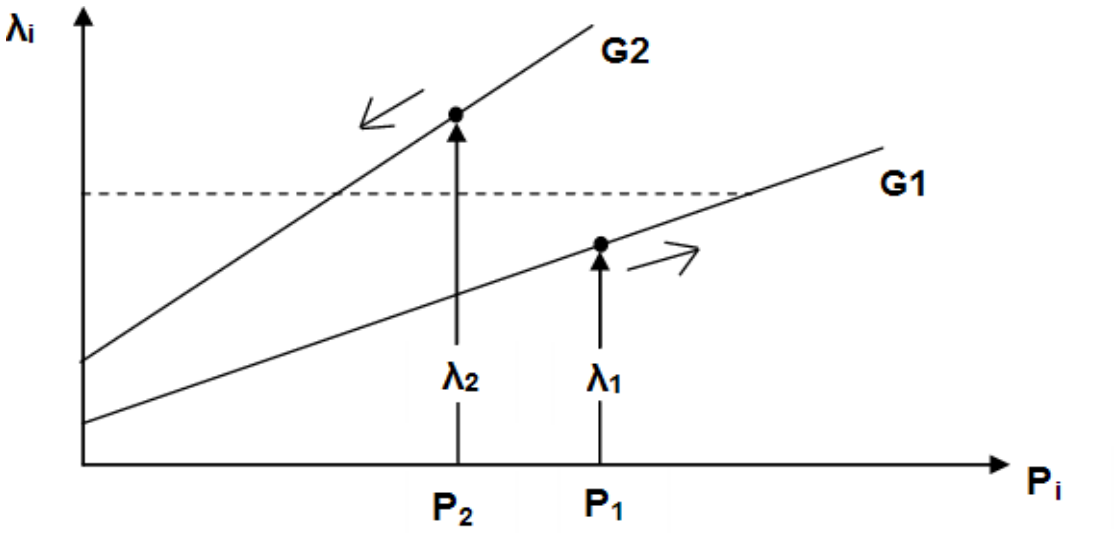
Και

$P_{loss}$: οι απώλειες μεταφοράς του συστήματος

Προκειμένου λοιπόν να λύσουμε το πρόβλημα της οικονομικής κατανομής φορτίου θα πρέπει να υπολογίσουμε την ενεργό ισχύ εξόδου κάθε γεννήτριας $P_i$ έτσι ώστε να ελαχιστοποιείται το συνολικό λειτουργικό κόστος του συστήματος $C_{total}$ αλλά να ικανοποιείται ο περιορισμός ισότητας (4.6) και οι ανισοτικοί περιορισμοί (4.1) .

Αντικειμενική συνάρτηση του συγκεκριμένου προβλήματος βελτιστοποίησης είναι η συνάρτηση του συνολικού κόστους του συστήματος $C_{total}$, μεταβλητές απόφασης είναι οι παραγωγή ισχύος εξόδου των γεννητριών.

Έστω ότι το σύστημά μας έχει δύο μονάδες παραγωγής με διαφορετικά οριακά κόστη $\lambda_1$ και $\lambda_2$ με $\lambda_2 > \lambda_1$ και διαφορετικές παραγωγές ισχύος $P_1$ και $P_2$. Αν υποθέσουμε ότι η γεννήτρια με το μεγαλύτερο οριακό κόστος (γεννήτρια 2) μειώσει τη παραγωγή της κατά $\Delta P$ και η γεννήτρια με το μικρότερο οριακό κόστος (γεννήτρια 1) αυξήσει τη παραγωγή της κατά $\Delta P$ τότε η μείωση του κόστους που θα πραγματοποιηθεί λόγω της μείωσης της παραγωγής της γεννήτριας με το μεγαλύτερο οριακό κόστος ($\Delta P \cdot \lambda_2$) θα είναι μεγαλύτερη από την αύξηση του κόστους που θα πραγματοποιηθεί από την αύξηση της παραγωγής της γεννήτριας με το μικρότερο οριακό κόστος ($\Delta P \cdot \lambda_1$), δηλαδή $\Delta P \cdot \lambda_2 > \Delta P \cdot \lambda_1$. Άρα το κόστος λειτουργίας του συστήματος θα μειωθεί. Αυτή η μείωση θα συνεχίζεται μέχρις ότου τα οριακά κόστη των δύο γεννητριών να γίνουν ίσα. Τότε θα έχουμε και τη βέλτιστη λειτουργία του συστήματος. Η περαιτέρω μείωση της παραγωγής της ισχύος της γεννήτριας 2 και η αύξηση της ισχύος της γεννήτριας 1 θα έχει πλέον ως αποτέλεσμα αύξηση του συνολικού κόστους του συστήματος μιας και πλέον θα ισχύει $\lambda_1 > \lambda_2$, άρα και $\Delta P \cdot \lambda_1 > \Delta P \cdot \lambda_2$ . Αυτό ισχύει και για περισσότερες από δύο γεννήτριες. Συμπερασματικά η βέλτιστη κατανομή φορτίου σε ένα σύστημα επιτυγχάνεται όταν όλες οι μονάδες λειτουργούν με το ίδιο οριακό κόστος [65]. Στο διάγραμμα 4.3 που ακολουθεί μπορούμε να δούμε και γραφικά αυτό που μόλις περιγράψαμε παραπάνω.




Διάγραμμα 4.3: Κατανομή της ισχύος σε ένα σύστημα με 2 γεννήτριες [64]

Όμως θα πρέπει να ικανοποιούνται και ο περιορισμός ισότητας (4.6) αλλά και οι ανισοτικοί περιορισμοί (4.1). Στην περίπτωση που όλες οι γεννήτριες λειτουργούν με το ίδιο οριακό κόστος χωρίς να παραβιάζουν τα όρια λειτουργίας τους τότε το πρόβλημα βελτιστοποίησης έχει λυθεί. Αν όμως κάποια γεννήτριας για το συγκεκριμένο οριακό κόστος παραβιάζει κάποιο όριό της, τότε η παραγωγή τίθεται ίση με το όριο το οποίο παραβιάζει και οι υπόλοιπες μονάδες μοιράζονται το υπολειπόμενο φορτίο με τον τρόπο που περιγράψαμε παραπάνω.

Τελικά έχουμε ότι κατά την επίλυση της οικονομικής κατανομής φορτίου ακολουθούμε τις εξής συνθήκες (Kuhn-Tucker) [68]:

$$P_{total} = \sum_{i=1}^{N} P_i = P_D + P_{loss}$$

$$\lambda_i = \frac{dC_i}{dP_i} = \lambda \text{ για κάθε } i = 1,2,\dots,N$$

$$P_{i,min} \le P_i \le P_{i,max} \text{ για κάθε } i = 1,2,\dots,N$$

$$\text{Αν } \frac{dC_i}{dP_i} < \lambda\text{: } P_i = P_{i,max}$$

$$\text{Αν } \frac{dC_i}{dP_i} > \lambda\text{: } P_i = P_{i,min}$$

Η παραπάνω μεθοδολογία επίλυσης της οικονομικής κατανομής φορτίου δεν συνυπολογίζει στο κόστος λειτουργίας κάθε μονάδας, το κόστος των απωλειών της γραμμής που την εξυπηρετεί. Αν για παράδειγμα μια μονάδα είναι ιδιαίτερα απομακρυσμένη από το σημείο κατανάλωσης θα έχει περισσότερες απώλειες σε σχέση με μια άλλη που βρίσκεται πιο κοντά στο σημείο κατανάλωσης και έτσι αν η πιο απομακρυσμένη μονάδα είναι αυτή με το μικρότερο οριακό κόστος μπορεί οι απώλειες μεταφοράς της γραμμής της να είναι τόσο μεγάλες που με αύξηση της ισχύος το συνολικό

κόστος να αυξηθεί αντί να μειωθεί [64],[66]. Στο πλαίσιο όμως αυτής της διπλωματικής αμελούμε το ζήτημα των απωλειών.

## 4.3 Μέθοδος πολλαπλασιαστών Lagrange αγνοώντας τις απώλειες μεταφοράς και τα όρια παραγωγής των μονάδων

Η μέθοδος πολλαπλασιαστών Lagrange ή αλλιώς μέθοδος συντελεστή λ είναι μία κλασσική μέθοδος επίλυσης του προβλήματος οικονομικής κατανομής φορτίου.

Αρχικά ορίζουμε την αντικειμενική συνάρτηση του προβλήματος βελτιστοποίησης που είναι η ακόλουθη [67],[68] :

$$minC_{total} = min\sum_{i=1}^{N} C_i = min\sum_{i=1}^{N}(a_i \cdot P_i^2 + b_i \cdot P_i + c_i) \quad (4.7)$$

$$s.t.\ P_D = \sum_{i=1}^{N} P_i \quad (4.8)$$

Στη συνέχεια σχηματίζουμε τη συνάρτηση Lagrange του παραπάνω προβλήματος :

$$L = C_{total} + \lambda \cdot \left(P_D - \sum_{i=1}^{N} P_i\right) = \sum_{i=1}^{N}(a_i \cdot P_i^2 + b_i \cdot P_i + c_i) + \lambda \cdot \left(P_D - \sum_{i=1}^{N} P_i\right) (4.9)$$

Όπου :

$\lambda$: ο πολλαπλασιαστής Lagrange σχετίζεται με τον περιορισμό ισότητας (4.8).

Το ελάχιστο της συνάρτησης Lagrange (4.9) βρίσκεται στο σημείο μηδενισμού όλων των μερικών παραγώγων της ως προς τις μεταβλητές $P_i$ για $i = 1,2,\dots,N$ και $\lambda$ [67],[68]. Έτσι λοιπόν έχουμε :

$$\frac{\partial L}{\partial P_i} = 0 \Rightarrow \frac{\partial C_i}{\partial P_i} - \lambda = 0 (4.10)$$

$$\frac{\partial L}{\partial \lambda} = 0\ (4.11)$$

Συνδυάζοντας την (4.9) με την (4.11) προκύπτει ο περιορισμός ισότητας (4.8). Επίσης συνδυάζοντας την (3.9) με την (3.10) προκύπτει η ακόλουθη σχέση :

$$2 \cdot \alpha_i \cdot P_i + b_i - \lambda = 0\ (4.12)$$

Λύνοντας την (4.12) ως προς $P_i$ προκύπτει η σχέση :

$$P_i = \frac{\lambda - b_i}{2 \cdot a_i}\ (4.13)$$

Αντικαθιστούμε την σχέση (4.13) στη σχέση (4.8) και λύνουμε ως προς $\lambda$ :

$$5P_D = \sum_{i=1}^{N} \frac{\lambda - b_i}{2 \cdot a_i} \Rightarrow P_D = \lambda \cdot \sum_{i=1}^{N} \frac{1}{2 \cdot \alpha_i} - \sum_{i=1}^{N} \frac{b_i}{2 \cdot a_i} \Rightarrow \lambda = \frac{P_D - \sum_{i=1}^{N} \frac{b_i}{2 \cdot a_i}}{\sum_{i=1}^{N} \frac{1}{2 \cdot a_i}} \quad (4.14)$$

Συγκεντρωτικά έχουμε ότι το για να λύσουμε το πρόβλημα οικονομικής κατανομής φορτίου με την μέθοδο των πολλαπλασιαστών Lagrange, αρχικά υπολογίζουμε τον πολλαπλασιαστή Lagrange ($€/MWh$) σύμφωνα με τη σχέση (4.14) και στη συνέχεια υπολογίζουμε την ενεργό παραγωγή κάθε γεννήτριας σύμφωνα με τη σχέση (4.13) [67].

Ο πολλαπλασιαστής Lagrange λ αποτελεί και το βέλτιστο οριακό κόστος. Αυτό γίνεται αντιληπτό από την σχέση (4.10) από την οποία προκύπτει ότι $\frac{\partial C_i}{\partial P_i} = \lambda$ για κάθε $i = 1,2, \dots, N$. Άρα η βέλτιστη τιμή του συνολικού λειτουργικού κόστους βρίσκεται στο σημείο όπου τα οριακά κόστη όλων των μονάδων είναι ίσα μεταξύ τους και ίσα με το πολλαπλασιαστή Lagrange [65].

### 4.3.1 Μέθοδος πολλαπλασιαστών Lagrange λαμβάνοντας υπόψη τα τεχνικά όρια των μονάδων

Αν τώρα θεωρήσουμε και τα όρια παραγωγής των γεννητριών τότε αφού λύσουμε το παραπάνω πρόβλημα και υπολογίσουμε όλες τις τιμές ενεργού ισχύος των γεννητριών ελέγχουμε αν βρίσκονται εντός των ορίων τους. Αν συμβαίνει αυτό τότε το πρόβλημα βελτιστοποίησης έχει λυθεί. Αν όμως δεν συμβαίνει αυτό και κάποιο όριο παραβιάζεται τότε η ισχύς εξόδου της εν λόγω γεννήτρια τίθεται ίση με το όριο που παραβιάζει (ή αν παραβιάζονται τα όρια σε παραπάνω από μία γεννήτριες τότε θέτουμε την παραγωγή κάθε μίας από αυτές ίση με το όριο που παραβιάζει) και λύνουμε τόσα προβλήματα βελτιστοποίησης όσα και οι περιορισμοί γεννητριών που παραβιάζονται. Στο τέλος βρίσκουμε για κάθε ένα από τα προβλήματα βελτιστοποίησης που λύσαμε το συνολικό κόστος λειτουργίας και συνεπώς η λύση του προβλήματος βελτιστοποίησης θα είναι αυτή που ελαχιστοποιεί το συνολικό ωραίο κόστος λειτουργίας [67].

# Κεφάλαιο 5

## *Βασικός αλγόριθμος*

## 5.1 Εισαγωγή στην ADMM μέθοδο

Καταρχάς πριν προχωρήσουμε στην ανάλυση της ADMM που χρησιμοποιείται στη διπλωματική για τη επίλυση του προβλήματος οικονομικής κατανομής φορτίου κρίνεται σκόπιμο να αναφερθούμε αρχικά σε δύο αλγορίθμους βελτιστοποίησης που αποτελούν τη βάση της.

### 5.1.1 Μέθοδος δυικής ανάβασης (Dual ascent)

Θεωρούμε το κυρτό πρόβλημα βελτιστοποίησης με περιορισμό ισότητας (equality-constrained convex optimization problem) [69] :

$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$

$$\text{s.to: } A \cdot \boldsymbol{x} = B \tag{5.1}$$

όπου $\boldsymbol{x} \epsilon \boldsymbol{R}^n$, $A \in \boldsymbol{R}^n$ και η $f: \boldsymbol{R}^n \rightarrow \boldsymbol{R}$ είναι κυρτή (convex) συνάρτηση.

Σχηματίζουμε τη συνάρτηση Lagrange του παραπάνω προβλήματος :

$$L(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x}) + \boldsymbol{y}^T \cdot (A \cdot \boldsymbol{x} - B) \quad (5.2)$$

Η δυική συνάρτηση (dual function) είναι:

$$g(\boldsymbol{y}) = \inf_{\boldsymbol{x},} L(\boldsymbol{x}, \boldsymbol{y}) = -f^*(-A^T \cdot \boldsymbol{y}) - B^T \cdot \boldsymbol{y} \ (5.3)$$

Όπου $\boldsymbol{y}$ είναι η δυική μεταβλητή (dual variable) ή ο πολλαπλασιαστής Lagrange και $f^*$ είναι η κυρτή συζυγής μιγαδική συνάρτηση (convex conjugate) της $f$.

Το δυικό πρόβλημα (dual problem) είναι το :

$$\max_{\boldsymbol{y}} g(\boldsymbol{y}) \ (5.4)$$

Όπου $\boldsymbol{y} \epsilon \boldsymbol{R}^n$. .

Αν υποθέσουμε ότι ισχύει αυστηρή δυικότητα (strong duality) τότε η βέλτιστες λύσεις του πρωτεύοντος (primal) και του δυικού (dual) προβλήματος είναι ίδιες [69],[70].

Μπορούμε να ανακτήσουμε ένα βέλτιστο σημείο $\boldsymbol{x}^*$από ένα δυικό βέλτιστο $\boldsymbol{y}^*$ ως εξής [70] :

$$\boldsymbol{x}^* = \underset{x}{argmin}\, L(\boldsymbol{x}, \boldsymbol{y}^*)$$

Προκειμένου να λύσουμε το δυικό πρόβλημα με τη δυική ανάβαση χρησιμοποιούμε τη μέθοδο της ανάβασης βαθμίδας (gradient ascent method) [69],[70].

Ορίζουμε $\nabla g(\boldsymbol{y})$ την κατεύθυνση της κίνησης με τη μεγαλύτερη αύξηση της $g(\boldsymbol{y})$, έτσι στόχος μας είναι να μεγιστοποιήσουμε την $g(\boldsymbol{y})$ κινούμενη προς αυτή τη κατεύθυνση [70]. Αν το $\nabla g(\boldsymbol{y})$ υπάρχει τότε [70]:

$$\nabla_y g(\boldsymbol{y}) = \nabla_y L(\boldsymbol{x}^*, y) = \nabla_y f(\boldsymbol{x}^*) + \nabla_y \boldsymbol{y}^T \cdot (A \cdot \boldsymbol{x} - B) = A \cdot \boldsymbol{x} - B$$

Έτσι λοιπόν η μέθοδος της δυικής ανάβασης απαρτίζεται από τα ακόλουθα βήματα:

$$\boldsymbol{x}(k+1) = \underset{x}{argmin}\, L\big(\boldsymbol{x}, \boldsymbol{y}(k)\big)$$

$$\boldsymbol{y}(k+1) = \boldsymbol{y}(k) + a(k) \cdot (A \cdot \boldsymbol{x}(k+1) - B) \tag{5.5}$$

Όπου $a(k)$: το βήμα αύξησης στην επανάληψη $k$.

Έτσι λοιπόν το πρώτο βήμα αυτού του αλγορίθμου είναι ελαχιστοποίηση ως προς $\boldsymbol{x}$ και το δεύτερο είναι ανανέωση της δυικής μεταβλητής $\boldsymbol{y}$. Εάν το $a(k)$ έχει επιλεγεί καταλλήλως και ισχύουν διάφορες άλλες ισχυρές υποθέσεις (η $f$ είναι αυστηρά κυρτή) τότε έχουμε σύγκλιση αμφότερων των $\boldsymbol{x}$ και $\boldsymbol{y}$**.** Ωστόσο το γεγονός ότι αυτές οι υποθέσεις δεν ισχύουν για πολλές εφαρμογές, μας αναγκάζει να βρούμε τρόπο ώστε να καλυτερεύσουμε τη μέθοδο.

### 5.1.2 Δυική αποσύνθεση (Dual decomposition)

Ας υποθέσουμε ότι η $f$ είναι διαχωρίσιμη (separable), δηλαδή ισχύει [69] :

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} f_i\,(x_i)$$

Όπου $\boldsymbol{x} = (x_1, x_2, \dots, x_N)$.

Αποσυνθέτουμε επίσης τον πίνακα A ως εξής :

$$A = [A_1, A_2, \dots, A_N]$$

Στη συνέχεια σχηματίζουμε τη συνάρτηση Lagrange :

$$L(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{N} L_i(x_i, y) = \sum_{i=1}^{N} (f_i(x_i) + \cdot\, \boldsymbol{y}^T A_i \cdot x_i - (1/N) \cdot \boldsymbol{y}^T \cdot B)$$

Παρατηρούμε ότι και η συνάρτηση Lagrange είναι διαχωρίσιμη σε $x$. Έτσι η ελαχιστοποίηση του $\boldsymbol{x}$ της σχέσης (5.5) μπορεί να χωριστεί σε N υποπροβλήματα τα οποία μπορούν να λύνονται παράλληλα. Έτσι λοιπόν ο αλγόριθμος έχει τα ακόλουθα βήματα :

$$x_i(k+1) = \underset{x_i}{argmin}\, L_i\big(x_i, \boldsymbol{y}(k)\big)$$

(5.6)

$$\boldsymbol{y}(k+1) = \boldsymbol{y}(k) + a(k) \cdot (A \cdot \boldsymbol{x}(k+1) - B)$$

Η διαφορά της δυικής αποσύνθεσης σε σχέση με τη δυική ανάβαση είναι ότι το πρόβλημα ελαχιστοποίησης ως προς $\boldsymbol{x}$ χωρίζεται σε N υποπροβλήματα που μπορούν να λυθούν ανεξάρτητα το ένα από το άλλο [70]. Η πορεία του αλγορίθμου είναι η ακόλουθη [69]: Κάθε επεξεργαστής (processor) υπολογίζει το δικό του $x_i(k+1)$ και στη συνέχεια το μεταδίδει (broadcast) σε όλους τους άλλους επεξεργαστές. Όταν ο κάθε επεξεργαστής συλλέξει όλα τα $x_i(k+1)$ και άρα μπορεί να σχηματίσει το διάνυσμα $\boldsymbol{x}(k+1)$, το χρησιμοποιεί ώστε να υπολογίζει τη καθολική (global) δυική μεταβλητή $\boldsymbol{y}(k+1)$. Μία διαφορετική προσέγγιση θα ήταν να οριστεί ένας κύριος επεξεργαστής (master processor), ο οποίος θα συλλέγει τα $x_i(k+1)$ από όλους τους άλλους επεξεργαστές και θα υπολογίζει έπειτα τη καθολική (global) δυική μεταβλητή $\boldsymbol{y}(k+1)$ και θα τη στέλνει πίσω στους άλλους επεξεργαστές ώστε να συνεχίσουν στην επόμενη επανάληψη [70].

### 5.1.3 Επαυξημένη (Augmented) συνάρτηση Lagrange και μέθοδος πολλαπλασιαστών Lagrange

Η μέθοδος επαυξημένης συνάρτησης Lagrange αναπτύχθηκε προκειμένου να κάνει την μέθοδο δυικής ανάβασης πιο "εύρωστη" (robust) και να έχουμε καλή σύγκλιση χωρίς προϋποθέσεις όπως είναι η αυστηρή κυρτότητα [69],[70].

Η επαυξημένη συνάρτηση Lagrange είναι :

$$L_\rho(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x}) + \boldsymbol{y}^T \cdot (A \cdot \boldsymbol{x} - B) + \frac{\rho}{2} \cdot \|A \cdot \boldsymbol{x} - B\|_2^2 \quad (5.7)$$

Όπου το $\rho > 0$ καλείται penalty.

Η συνάρτηση $L_\rho$ είναι η συνάρτηση Lagrange του παρακάτω προβλήματος :

$$\min_{\boldsymbol{x},} f(\boldsymbol{x}) + \frac{\rho}{2} \cdot \|A \cdot \boldsymbol{x} - B\|_2^2$$

(5.8)

$$s.to:\ A \cdot \boldsymbol{x} = B$$

Το εν λόγω πρόβλημα είναι ισοδύναμο με το αρχικό πρόβλημα (5.1). Αυτό ισχύει διότι για κάθε $\boldsymbol{x}$ που είναι πιθανή λύση του προβλήματος ο όρος $\frac{\rho}{2} \cdot \|A \cdot \boldsymbol{x} - B\|_2^2$ είναι ίσος με μηδέν.

Εφαρμόζουμε λοιπόν τη μέθοδο δυικής ανάβασης στο πρόβλημα που δημιουργήσαμε και προκύπτουν :

$$\boldsymbol{x}(k+1) = \underset{x}{argmin}\, L_p\big(\boldsymbol{x}, \boldsymbol{y}(k)\big)$$

$$\boldsymbol{y}(k+1) = \boldsymbol{y}(k) + \rho \cdot (A \cdot \boldsymbol{x}(k+1) - B) \tag{5.9}$$

Η μέθοδος που περιγράφηκε στη συγκεκριμένη ενότητα ονομάζεται μέθοδος πολλαπλασιαστών Lagrange (method of Lagrange multipliers). Σε αυτή τη μέθοδο χρησιμοποιούμε τη παράμετρο $\rho$ ως βήμα αύξησης της δυικής μεταβλητής $\boldsymbol{y}$. Ο όρος $\frac{\rho}{2} \cdot \left|\left|A \cdot \boldsymbol{x} - B\right|\right|_2^2$ προστίθεται έτσι ώστε ο αλγόριθμος να συγκλίνει κάτω από πιο χαλαρές συνθήκες σε σχέση με αυτές που απαιτεί η μέθοδος δυικής ανάβασης. Βέβαια έχει το μειονέκτημα ότι με την προσθήκη αυτού του τετραγωνικού όρου δεν μπορούμε να εφαρμόσουμε τη μέθοδο δυικής αποσύνθεσης [69],[70]. Στην ενότητα που ακολουθεί θα δούμε πως μπορούμε να άρουμε αυτό τον περιορισμό με τη χρήση της μεθόδου ADMM.

### 5.1.4 ADMM

Η μέθοδος ADMM **(**Alternating Directional Method of Multipliers) δημιουργήθηκε ώστε να συνδυάσει τη δυνατότητα αποσύνθεσης της δυικής ανάβασης με τις ανώτερες ιδιότητες σύγκλισης της μεθόδου των πολλαπλασιαστών Lagrange [69],[70]. Θεωρούμε ένα πρόβλημα ελαχιστοποίησης με διαχωρίσιμες αντικειμενικές συναρτήσεις και περιορισμούς στα διανύσματα $\boldsymbol{x}$ και $z$ [71]:

$$\min_{\boldsymbol{x},\boldsymbol{z}} f(\boldsymbol{x}) + g(\boldsymbol{z})$$

$$\text{s.to} : \boldsymbol{x} \in K_x, \boldsymbol{z} \in K_z \tag{5.10}$$

$$A \cdot \boldsymbol{x} + B \cdot \boldsymbol{z} = c$$

Σχηματίζουμε την επαυξημένη συνάρτηση Lagrange (Augmented Lagrangian):

$$L_\rho(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\xi}) = f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{\xi}^T \cdot (A \cdot \boldsymbol{x} + B \cdot \boldsymbol{z} - c) + \frac{\rho}{2} \cdot \|A \cdot \boldsymbol{x} + B \cdot \boldsymbol{z} - c\|_2^2 \tag{5.11}$$

Στη συνέχεια το πρόβλημα αποσυντίθεται στη γενική μορφή της ADMM [9]:

$$\boldsymbol{x}(k+1) = \underset{x \in K_x}{argmin}\, L_\rho\big(\boldsymbol{x}, \boldsymbol{z}(k), \boldsymbol{\xi}(k)\big) \tag{5.12}$$

$$\boldsymbol{z}(k+1) = \underset{z \in K_z}{argmin}\, L_\rho\big(\boldsymbol{x}(k+1), \boldsymbol{z}, \boldsymbol{\xi}(k)\big) \tag{5.13}$$

$$\boldsymbol{\xi}(k+1) = \boldsymbol{\xi}(k) + \rho \cdot (A \cdot \boldsymbol{x}(k+1) + B \cdot \boldsymbol{z}(k+1) - c) \tag{5.14}$$

Η ADMM μέθοδος ουσιαστικά περιλαμβάνει τα ακόλουθα στάδια: Αρχικά κάνουμε μία ελαχιστοποίηση ως προς $\boldsymbol{x}$, μετά ελαχιστοποιούμε ως προς $\boldsymbol{z}$ και τέλος αναβαθμί-

ζουμε τη δυική μεταβλητή με βήμα αύξησης $\rho$, όπως και στη μέθοδο των πολλαπλασιαστών Lagrange [69].

## 5.2 Αλγόριθμος που βασίζεται στη μέθοδο ADMM για τη λύση του προβλήματος Οικονομικής Κατανομής Φορτίου

Το πρόβλημα Οικονομικής Κατανομής Φορτίου περιγράφηκε στο κεφάλαιο 4. Το αναφέρουμε ξανά συνοπτικά ακολουθώντας το συμβολισμό που υπάρχει στο [72]:

$$min\, f(\boldsymbol{x}) = min \sum_{i=1}^{N} f_i(x_i) \quad (5.15)$$

Όπου:

$f_i(x_i)$ : η συνάρτηση κόστους της κάθε γεννήτριας $i$

και

$x_i$ : η ενεργός ισχύς εξόδου της κάθε γεννήτριας $i$.

Στη δικιά μας περίπτωση :

$$f_i(x_i) = a_i \cdot x_i^2 + b_i \cdot x_i + c_i$$

Οι περιορισμοί του προβλήματος (5.15) είναι οι ακόλουθοι :

$$x_i^{min} \le x_i \le x_i^{max} \text{ για κάθε } i = 1,2, \dots, N \quad (5.16)$$

Όπου $x_i^{min}$ και $x_i^{max}$ το τεχνικό ελάχιστο και μέγιστο αντίστοιχα της κάθε γεννήτριας $i$.

$$\sum_{i=1}^{N} x_i = P_d (5.17)$$

**<u>Υπόθεση 1</u>** : Για κάθε $i = 1,2, \dots, N$ η συνάρτηση κόστους $f_i(.)$: $R_+ \to R_+$ είναι αυστηρά κυρτή και συνεχώς παραγωγίσιμη, όπου το $R_+$ δηλώνει σύνολο από μη αρνητικούς πραγματικούς αριθμούς.

Η ανάλυση του αλγορίθμου η οποία περιγράφεται και στο [72] είναι η ακόλουθη :

Προκειμένου να χρησιμοποιήσουμε την ADMM μέθοδο για να λύσουμε το πρόβλημα Οικονομικής Κατανομής φορτίου πρέπει να μετασχηματίσουμε το πρόβλημα (5.15) –(5.17). Αρχικά ορίζουμε δύο κυρτά (convex) σύνολα :

$$\boldsymbol{x} = (x_1, x_2, \dots, x_N)^T$$

Και

$$\boldsymbol{y} = (y_1, y_2, \dots, y_N)^T$$

Κατόπιν ορίζουμε τα σύνολα :

$$\Omega_1 = \left\{ \boldsymbol{x} \in R^N \middle| x_i^{min} \le x_i \le x_i^{max} \ \gamma\iota\alpha \ \forall \ i = 1,2,N \right\} (5.18)$$

$$\Omega_2 = \left\{ \boldsymbol{y} \in R^N \middle| \sum_{i=1}^{N} \mathrm{y_i} = \mathrm{P_d} \right\} (5.19)$$

Επίσης ορίζουμε δύο δεικτικές συναρτήσεις (indicator functions) $g_1(\boldsymbol{x})$ και $g_2(\boldsymbol{y})$ για κάθε σύνολο $\Omega_1$ και $\Omega_2$ :

$$g_1(\boldsymbol{x}) = \begin{cases} 0 \ , \alpha\nu \ \boldsymbol{x} \in \Omega_1 \\ +\infty \ , \delta\iota\alpha\varphi o\rho\varepsilon\tau\kappa\acute{\alpha} \end{cases} (5.20)$$

$$g_2(\boldsymbol{y}) = \begin{cases} 0 \ , \alpha\nu \ \boldsymbol{y} \in \Omega_2 \\ +\infty \ , \delta\iota\alpha\varphi o\rho\varepsilon\tau\kappa\acute{\alpha} \end{cases} (5.21)$$

Έτσι το πρόβλημα Οικονομικής Κατανομής φορτίου μετασχηματίζεται :

$$min[f(\boldsymbol{x}) + g_1(\boldsymbol{x}) + g_2(\boldsymbol{y})]$$

(5.22)

$$\text{Subject to : } \boldsymbol{x} - \boldsymbol{y} = 0 \ \mu\varepsilon \ \boldsymbol{x}, \boldsymbol{y} \in R^N$$

Σχηματίζουμε την επαυξημένη συνάρτηση Lagrange (Augmented Lagrangian) του προβλήματος (5.22) :

$$L_\rho(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + g_1(\boldsymbol{x}) + g_2(\boldsymbol{y}) + \boldsymbol{\lambda}^T \cdot (\boldsymbol{x} - \boldsymbol{y}) + \frac{\rho}{2} \cdot \|\boldsymbol{x} - \boldsymbol{y}\|_2^2 (5.23)$$

Οι λύσεις της (5.23) είναι οι ακόλουθες :

$$\boldsymbol{x}(k+1) = \underset{x}{argmin} \, L_\rho\big(\boldsymbol{x}, \boldsymbol{y}(k), \boldsymbol{\lambda}(k)\big) (5.24)$$

$$\boldsymbol{y}(k+1) = \underset{y}{argmin} \, L_\rho\big(\boldsymbol{x}(k+1), \boldsymbol{y}, \boldsymbol{\lambda}(k)\big) (5.25)$$

$$\boldsymbol{\lambda}(k+1) = \boldsymbol{\lambda}(k) + \rho \cdot \big(\boldsymbol{x}(k+1) - \boldsymbol{y}(k+1)\big) (5.26)$$

**<u>Υπόθεση 2</u>**: Η μη επαυξημένη συνάρτηση Lagrange $L_0(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + g_1(\boldsymbol{x}) + g_2(\boldsymbol{y}) + \boldsymbol{\lambda}^T \cdot (\boldsymbol{x} - \boldsymbol{y})$ έχει ένα σαγματικό σημείο (saddle point) π.χ. υπάρχει μία βέλτιστη λύση $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{\lambda}^*)$ τέτοια ώστε:

$$L_0(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{\lambda}) \le L_0(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{\lambda}^*) \le L_0(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\lambda}^*) (5.27)$$

Που ισχύει για όλα τα $\boldsymbol{x} \in \Omega_1$ , $\boldsymbol{y} \in \Omega_2$ και $\boldsymbol{\lambda} \in R^N$

**Ορισμός 1** :Τα primal και dual residual ορίζονται αντίστοιχα :

$$\boldsymbol{r}(k) = \boldsymbol{x}(k) - \boldsymbol{y}(k) \text{ (5.28)}$$

$$\boldsymbol{s}(k) = -\rho \cdot \big(\boldsymbol{y}(k) - \boldsymbol{y}(k-1)\big) \text{ (5.29)}$$

**Λήμμα 1**: Αν ισχύει η υπόθεση 2 και $\rho \geq 0$ τότε οι επαναλήψεις (5.24) -(5.26) μπορούν να συγκλίνουν στη βέλτιστη λύση $\boldsymbol{x}^*, \boldsymbol{y}^*$ και το βέλτιστο πολλαπλασιαστή Lagrange του προβλήματος (5.22) όταν :

$$\lim_{k\to+\infty} \|\boldsymbol{r}(k)\| = 0 \; και \; \lim_{k\to+\infty} \|\boldsymbol{s}(k)\| = 0 \text{ (5.30)}$$

Αυτό ισοδύναμα σημαίνει ότι για να έχουμε σύγκλιση θα πρέπει να ισχύει :

$$|r_i(k)| \leq \varepsilon \Rightarrow |x_i(k) - y_i(k)| \leq \varepsilon \; για \; κάθε \; i = 1,2,\dots,N \text{ (5.31)}$$

$$|s_i(k)| \leq \varepsilon \Rightarrow |-\rho \cdot (y_i(k) - y_i(k-1))| \leq \varepsilon \; για \; κάθε \; i = 1,2,\dots,N \text{ (5.32)}$$

Όπου $\varepsilon$: πολύ μικρός αριθμός

Αντικαθιστώντας την σχέση (5.23) στην σχέση (5.24) λαμβάνουμε :

$$\boldsymbol{x}(k+1) = \underset{x}{argmin}\, L_p\big(\boldsymbol{x}, \boldsymbol{y}(k), \boldsymbol{\lambda}(k)\big) =$$

$$= \underset{x}{argmin}\, f(\boldsymbol{x}) + g_1(\boldsymbol{x}) + g_2(\boldsymbol{y}) + \boldsymbol{\lambda}^T(k) \cdot (\boldsymbol{x} - \boldsymbol{y}(k)) + \frac{p}{2} \cdot || \, \boldsymbol{x} - \boldsymbol{y}(k) \, ||_2^2 =$$

$$= \underset{x\in\Omega_1}{argmin}\, f(\boldsymbol{x}) + \boldsymbol{\lambda}^T(k) \cdot (\boldsymbol{x} - \boldsymbol{y}(k)) + \frac{p}{2} \cdot || \, \boldsymbol{x} - \boldsymbol{y}(k) \, ||_2^2 =$$

$$= \underset{x\in\Omega_1}{argmin} \sum_{i=1}^{N} f_i(x_i) + \sum_{i=1}^{N} \lambda_i(k) \cdot (x_i - y_i(k)) + \frac{p}{2} \cdot \sum_{i=1}^{N} (x_i - y_i(k))^2 =$$

$$\underset{x\in\Omega_1}{argmin} \sum_{i=1}^{N} \Big[ f_i(x_i) + \lambda_i(k) \cdot (x_i - y_i(k)) + \frac{p}{2} \cdot (x_i - y_i(k))^2 \Big]$$

με $f_i(x_i) = a_i \cdot x_i^2 + b_i \cdot x_i + c_i$

Έστω $C_i(x_i) = f_i(x_i) + \lambda_i(k) \cdot (x_i - y_i(k)) + \frac{p}{2} \cdot (x_i - y_i(k))^2$

Λύνουμε λοιπόν το πρόβλημα :

$$min \sum_{i=1}^{N} C_i(x_i)$$

(5.33)

$$\text{s.t. } x_i^{min} \leq x_i \leq x_i^{max} \text{ για κάθε } i = 1,2,\dots,N$$

Οι λύσεις του προβλήματος (5.33) είναι η ακόλουθες :

$$\frac{\partial C_i(x_i)}{\partial x_i} = 0 \Rightarrow$$

$$2 \cdot a_i \cdot x_i + b_i + \lambda_i(k) + \rho \cdot \big(x_i - y_i(k)\big) = 0 \Rightarrow$$

$$x_i(k+1) = \frac{-b_i - \lambda_i(k) + p \cdot y_i(k)}{2a_i + p}$$

Όμως πρέπει να ελέγξω ότι η κάθε λύση είναι εντός του αντίστοιχου τεχνικού ελαχίστου και μεγίστου της.

Τελικά:

$$x_i(k+1) = min\left\{max\left\{\frac{-b_i - \lambda_i(k) + p \cdot y_i(k)}{2a_i + p}, x_i^{min}\right\}, x_i^{max}\right\} \quad (5.34)$$

Κατά όμοιο τρόπο αντικαθιστώντας την (5.23) στην (5.25) λαμβάνουμε :

$$\boldsymbol{y}(k+1) = \underset{y}{\text{argmin}}\, L_p\big(\boldsymbol{x}(k+1), \boldsymbol{y}, \boldsymbol{\lambda}(k)\big) =$$

$$= \underset{y}{argmin}\, f(\boldsymbol{x}) + g_1(\boldsymbol{x}) + g_2(\boldsymbol{y}) + \boldsymbol{\lambda}^T(k) \cdot (\boldsymbol{x}(k+1) - \boldsymbol{y}) + \frac{p}{2} \cdot || \boldsymbol{x}(k+1) - \boldsymbol{y} ||_2^2 =$$

$$= \underset{y}{argmin} g_2(\boldsymbol{y}) + \boldsymbol{\lambda}^T(k) \cdot (\boldsymbol{x}(k+1) - \boldsymbol{y}) + \frac{p}{2} \cdot || \boldsymbol{x}(k+1) - \boldsymbol{y} ||_2^2 =$$

$$= \underset{y \in \Omega_2}{argmin}\, \boldsymbol{\lambda}^T(k) \cdot (\boldsymbol{x}(k+1) - \boldsymbol{y}) + \frac{p}{2} \cdot || \boldsymbol{x}(k+1) - \boldsymbol{y} ||_2^2 =$$

$$= \underset{y \in \Omega_2}{argmin} \sum_{i=1}^{N} \lambda_i(k) \cdot (x_i(k+1) - y_i) + \frac{p}{2} \cdot \sum_{i=1}^{N} (x_i(k+1) - y_i)^2$$

Το παραπάνω πρόβλημα μπορεί να μετασχηματιστεί ως εξής :

$$min \sum_{i=1}^{N} \left[ \lambda_i(k) \cdot (x_i(k+1) - y_i) + \frac{p}{2} \cdot (x_i(k+1) - y_i)^2 \right]$$

(5.35)

$$s.t. \sum_{i=1}^{N} \mathrm{y_i} = \mathrm{P_d}$$

Χρησιμοποιώντας τη μέθοδο των πολλαπλασιαστών Lagrange, μετασχηματίζουμε το πρόβλημα (5.35) σε ένα πρόβλημα βελτιστοποίησης χωρίς περιορισμούς, που είναι το ακόλουθο :

$$L = \sum_{i=1}^{N} \left[ \lambda_i(k) \cdot (x_i(k+1) - y_i) + \frac{p}{2} \cdot (y_i - x_i(k+1))^2 \right] + \eta \cdot \left( P_d - \sum_{i=1}^{N} y_i \right)$$

Όπου $\eta$: ο πολλαπλασιαστής Lagrange που σχετίζεται με τον περιορισμό ισότητας του προβλήματος (5.35).

Για κάθε κόμβο $i$ η λύση του προβλήματος (5.35) βρίσκεται ως ακολούθως :

$$\frac{\partial L}{\partial y_i} = 0 \Rightarrow -\lambda_i(k) + p \cdot \left( y_i - x_i(k+1) \right) - \eta = 0 \text{ (5.36)}$$

$$\frac{\partial L}{\partial \eta} = 0 \Rightarrow P_d - \sum_{i=1}^{N} y_i = 0 \text{ (5.37)}$$

Από τη σχέση (5.36) λύνουμε ως προς $y_i$ και έχουμε :

$$y_i = \frac{1}{\rho} \cdot (\lambda_i(k) + \eta) + x_i(k+1) \text{(5.38)}$$

Στη συνέχεια αντικαθιστούμε την σχέση (5.38) στη σχέση (5.37) και λύνοντας ως προς $\eta$ έχουμε:

$$\sum_{i=0}^{N} \left[ \frac{1}{\rho} \cdot (\lambda_i(k) + \eta) + x_i(k+1) \right] = P_d \Rightarrow$$

$$\sum_{i=0}^{N} \frac{\lambda_i(k)}{\rho} + N \cdot \frac{\eta}{\rho} + \sum_{i=0}^{N} x_i(k+1) = P_d \Rightarrow$$

$$\eta = \frac{\rho}{N} \cdot \left( P_d - \sum_{i=1}^{N} \left[ x_i(k+1) + \frac{\lambda_i(k)}{\rho} \right] \right) \text{ (5.39)}$$

Άρα η λύση του προβλήματος (5.35) δίνεται από την ακόλουθη σχέση :

$$y_i(k+1) = \frac{\eta}{\rho} + \frac{\lambda_i(k)}{\rho} + x_i(k+1) \quad (5.38)$$

Όπου το $\eta$ δίνεται από τη σχέση (5.39).

Τέλος αντικαθιστώντας τα $x_i(k+1)$ και $y_i(k+1)$ από τις σχέσεις (5.34) και (5.38) αντίστοιχα στη σχέση (5.26), ανανεώνουμε το πολλαπλασιαστή Lagrange σύμφωνα με τη σχέση :

$$\lambda_i(k+1) = \lambda_i(k) + \rho \cdot \left(x_i(k+1) - y_i(k+1)\right) \quad (5.40)$$

Από το Λήμμα 1 έχουμε ότι η βέλτιστη λύση του προβλήματος Οικονομικής Κατανομής Φορτίου (5.15)-(5.17), επιτυγχάνεται στην επανάληψη $k$ στην οποία ισχύουν οι σχέσεις (5.31) και (5.32).

Συνοπτικά ο αλγόριθμος που λύνει το πρόβλημα Οικονομικής Κατανομής φορτίου παρουσιάζεται παρακάτω:

*Input:* $\rho, x_i^{min}, x_i^{max}, x_i(0), y_i(0), \lambda_i(0), P_d$ *for* $i = 1,2, \dots, N$

*Output: Optimal power generation* $\boldsymbol{x}^*$

*repeat:*

*for* $i = 1,2, \dots, N$ *do :*

$$x_i(k+1) = min\left\{max\left\{\frac{-b_i - \lambda_i(k) + p \cdot y_i(k)}{2a_i + p}, x_i^{min}\right\}, x_i^{max}\right\}$$

end for

*for* $i = 1,2, \dots, N$ *do :*

$$\eta = \frac{\rho}{N} \cdot \left(P_d - \sum_{i=1}^{N}\left[x_i(k+1) + \frac{\lambda_i(k)}{\rho}\right]\right)$$

$$y_i(k+1) = \frac{\eta}{\rho} + \frac{\lambda_i(k)}{\rho} + x_i(k+1)$$

$$\lambda_i(k+1) = \lambda_i(k) + p \cdot \left(x_i(k+1) - y_i(k+1)\right)$$

*end for*

if $(|r_i(k+1))| < \varepsilon, |s_i(k+1)| < \varepsilon \; for \; i = 1,2, \dots, N)$ then

return $x^*$

end if

until $(|r_i(k+1))| < \varepsilon, |s_i(k+1)| < \varepsilon \; for \; i = 1,2, \dots, N)$

Όπως παρατηρούμε ο υπολογισμός της πρωτεύουσας μεταβλητής $y_i(k+1)$ γίνεται με κεντρικό τρόπο μιας και για τον υπολογισμό του βέλτιστου πολλαπλασιαστή $\eta$ απαιτείται καθολική πληροφορία. Συγκεκριμένα για να λυθεί το πρόβλημα (5.35) και να υπολογιστούν οι λύσεις $y_i(k+1)$ αρχικά θα πρέπει να υπολογιστεί το $\eta$ από τη σχέση (5.39) που απαιτεί τη γνώση του αθροίσματος $\sum_{i=1}^{N}\left[x_i(k+1)+\frac{\lambda_i(k)}{p}\right]$. Δηλαδή θα πρέπει να προστεθούν οι λύσεις $x_i(k+1)$ όλων των γεννητριών του συστήματος. Έτσι λοιπόν απαιτείται καθολική πληροφορία (global information), γεγονός που κάνει τον αλγόριθμο κεντρικό.

## 5.3 Προτεινόμενος αποκεντρωμένος αλγόριθμος

Θεωρούμε ένα δίκτυο όπου υπάρχουν κάποιοι κόμβοι με γεννήτριες. Θέλουμε με δεδομένο φορτίο συστήματος και όρια παραγωγής των γεννητριών να λύσουμε το πρόβλημα οικονομικής κατανομής φορτίου κάνοντας χρήση του αλγορίθμου που αναπτύξαμε στην ενότητα 5.2. Ζητάμε δηλαδή την παραγωγή ενεργού ισχύος κάθε γεννήτριας έτσι ώστε να ικανοποιείται το φορτίο του συστήματος μας, να μην παραβιάζονται τα όρια παραγωγής των γεννητριών και συνάμα το συνολικό λειτουργικό ωραίο κόστος παραγωγής να είναι ελάχιστο. Σε κάθε κόμβο του δικτύου θεωρούμε ότι υπάρχει ένας μικροϋπολογιστής (π.χ. Raspberry), ο οποίος λύνει ένα πρόβλημα βελτιστοποίησης της μορφής (5.33):

$$min\, C_i(x_i) = \min\left(f_i(x_i) + \lambda_i(k)\cdot(x_i - y_i(k)) + \frac{p}{2}\cdot(x_i - y_i(k))^2\right)$$

$$\text{s.to}: x_i^{min} \le x_i \le x_i^{max}$$

Όταν ο κόμβος $i$ υπολογίσει τη λύση του $x_i(k+1)$ τη στέλνει στο τοπικό blockchain στο οποίο είναι μέλος. Στο blockchain υπάρχει καταχωρημένο ένα έξυπνο συμβόλαιο το οποίο αφού λάβει τις τοπικές λύσεις από όλους κόμβους λύνει ένα πρόβλημα βελτιστοποίησης της μορφής (5.35) :

$$min\sum_{i=1}^{N}\left[\lambda_i(k)\cdot(x_i(k+1)-y_i) + \frac{p}{2}\cdot(x_i(k+1)-y_i)^2\right]$$

$$s.to: \sum_{i=1}^{N} \mathrm{y_i} = \mathrm{P_d}$$

Αφού υπολογίσει τις λύσεις $y_i(k+1)$ υπολογίζει την ανανέωση των πολλαπλασιαστών Lagrange $\lambda_i(k+1)$ σύμφωνα με τη σχέση (5.40) :

$$\lambda_i(k+1) = \lambda_i(k) + p\cdot\left(x_i(k+1) - y_i(k+1)\right)$$

Τέλος ελέγχει αν έχει επέλθει σύγκλιση έτσι ώστε να ενημερώσει τους κόμβους ότι το πρόβλημα βελτιστοποίησης έχει τελειώσει.

Συγκεντρωτικά αυτό που κάνει το έξυπνο συμβόλαιο είναι :

Δέχεται από όλους τους κόμβους τις λύσεις τους $x_i(k+1)$ που σχετίζονται με τα τοπικά όρια παραγωγής τους και υπολογίζει τις τιμή της ισχύος $y_i(k+1)$ που θα πρέπει να παράγει κάθε κόμβος έτσι ώστε το συνολικό φορτίο του συστήματος να ικανοποιείται. Στη συνέχεια κάθε κόμβος λαμβάνει το $y_i(k+1)$ και λύνει ξανά. Άρα το συμβόλαιο στο blockchain θέλει να συνδυάσει όλα τα τοπικά προβλήματα βελτιστοποίησης ούτως ώστε να φτάσουμε στη βέλτιστη τιμή του συστήματος (global optimum). Το blockchain έχει το ρόλο του σωρευτή (aggregator) που παρέχει στους κόμβους ένα βήμα σώρευσης δεδομένων. Τους παρέχει δηλαδή μία τιμή ισχύος που με βάση αυτή ο κάθε κόμβος θα πρέπει να βρει μια δικιά του τιμή ισχύος που είναι εντός των ορίων παραγωγής του, αλλά θα πρέπει να προσεγγίσει και την τιμή που έλαβε από το έξυπνο συμβόλαιο. Όταν αυτό συμβεί τότε θα έχει επιτευχθεί σύγκλιση. Μια παρόμοια υλοποίηση έχει γίνει στο [71], με τη διαφορά ότι στην εν λόγω λύνεται το πρόβλημα βέλτιστης ροής ισχύος (Optimal power Flow Problem) σε ένα πιο περίπλοκο σύστημα.

Επιλέξαμε να πραγματοποιήσουμε το blockchain σαν το μέσο στο οποίο θα γίνεται το aggregation step μιας και είναι μια πλατφόρμα όπου δεν χρειάζεται να υπάρχει εμπιστοσύνη μεταξύ των συμμετεχόντων. Το blockchain εγγυάται δίκαιο και σωστό υπολογισμό του aggregation step, ελέγχει αν έχει υπάρχει σύγκλιση και αποθηκεύει τις τιμές ισχύος που θα πρέπει να παράγει κάθε γεννήτρια. Και όλες αυτές οι διαδικασίες γίνονται με διαφάνεια και μπορούν να επιβεβαιωθούν από όλους τους συμμετέχοντες.

Ακολουθεί ένα σχεδιάγραμμα το οποίο παρουσιάζει συνοπτικά τη λειτουργία του αποκεντρωμένου αλγορίθμου :

repeat

- *Locally computation for* $i = 1,2,\dots,N$

$$x_i(k+1) = min\left\{max\left\{\frac{-b_i - \lambda_i(k) + p \cdot y_i(k)}{2a_i + p}, x_i^{min}\right\}, x_i^{max}\right\}$$

- *ADMM aggregator, on blockchain*
  *Gather all* $x_i(k+1)$ *and compute :*

$$\eta = \frac{\rho}{N} \cdot \left(P_d - \sum_{i=1}^{N}\left[x_i(k+1) + \frac{\lambda_i(k)}{\rho}\right]\right)$$

$$y_i(k+1) = \frac{\eta}{\rho} + \frac{\lambda_i(k)}{\rho} + x_i(k+1)$$

$$\lambda_i(k+1) = \lambda_i(k) + p \cdot \left(x_i(k+1) - y_i(k+1)\right)$$

if $(|r_i(k+1))| < \varepsilon, |s_i(k+1)| < \varepsilon \; for \; i = 1,2,\dots,N)$ then

return $x^*$

end

until $(|r_i(k+1))| < \varepsilon, |s_i(k+1)| < \varepsilon \; for \; i = 1,2,\dots,N)$

Στο κεφάλαιο 6 που ακολουθεί παρατίθενται συνοπτικά τα εργαλεία τα οποία χρησιμοποιήθηκαν για την ανάπτυξη αυτής της εφαρμογής.

# Κεφάλαιο 6

## *Εργαλεία και τεχνολογίες*

## 6.1 Solidity

Η solidity [73] είναι μία αντικειμενοστραφής (object-oriented) γλώσσα προγραμματισμού υψηλού επιπέδου η οποία είναι επηρεασμένη από άλλες γλώσσες όπως η C++, η python και η Javascript. Είναι σχεδιασμένη ώστε να τρέχει στην Εικονική Μηχανή του Ethereum και είναι ιδανική για τη δημιουργία των έξυπνων συμβολαίων (smart contracts). Λόγω του γεγονότος ότι μοιάζει αρκετά με τη Javascript είναι κατανοητή από αρκετούς προγραμματιστές [74]. Επίσης είναι γλώσσα στατικού τύπου και υποστηρίζει την κληρονομικότητα (inheritance), τις βιβλιοθήκες και πολύπλοκους τύπους που ορίζονται από τον προγραμματιστή. Η solidity μεταγλωττίζεται σε bytecode που είναι εκτελέσιμο από την εικονική μηχανή του Ethereun (Ethereum Virtual Machine). Με τη solidity μπορούν να δημιουργηθούν έξυπνα συμβόλαια με σκοπό τη συγκέντρωση χρημάτων για φιλανθρωπικό σκοπό, διαφόρων ειδών δημοπρασίες καθώς και πορτοφόλια πολλαπλών υπογραφών [73]. Ουσιαστικά η solidity χρησιμοποιήθηκε στα πλαίσια αυτής της διπλωματικής για την υλοποίηση του έξυπνου συμβολαίου που αναπτύσσεται στο blockchain και δημιουργεί το aggregator step που χρειάζεται η ADMM μέθοδος.

## 6.2 Remix

Το Remix [75] είναι ένα ισχυρό εργαλείο ανοιχτού κώδικα, που μας επιτρέπει να γράψουμε και να μεταγλωττίσουμε έξυπνα συμβόλαια. Είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE-Integrated Development Environment) κώδικα γραμμένο σε Javascript και μπορεί να χρησιμοποιηθεί είτε τοπικά στον υπολογιστή είτε απευθείας στο φυλλομετρητή (browser). Μέσω του Remix μπορούν να γίνουν μεταγλώττιση (compile) τα συμβόλαια, να αναπτυχθούν (deployment) στο blockchain και επίσης υποστηρίζει αποσφαλμάτωση (debugging). Επίσης επιτρέπει στο χρήστη να κάνει διάφορα tests και γενικά είναι αρκετά εύκολο στη χρήση του. Στην παρούσα διπλωματική χρησιμοποιήθηκε το Remix για την ανάπτυξη του έξυπνου συμβολαίου και για να γίνει το deployment στο ganache. Επίσης γίνανε κάποια tests για τη διερεύνηση της εγκυρότητας της εκτέλεσης του έξυπνου συμβολαίου.

## 6.3 Ganache Gui

Το Ganache [76] είναι ένα τοπικό προσομοιωμένο Ethereum blockchain το οποίο μπορεί να χρησιμοποιηθεί για να αναπτυχθούν τα έξυπνα συμβόλαια και να εκτελεστούν διάφορα σενάρια. Μπορεί να δημιουργήσει μέχρι και 100 διαφορετικούς λογαριασμούς (με 100 ether έκαστος), επιτρέπει στο χρήστη να έχει πρόσβαση σε events, στην κατανάλωση gas κάθε λογαριασμού κατά την κλήση συναρτήσεων και επίσης το mining γίνεται άμεσα. Όλα αυτά σε συνδυασμό με το γεγονός ότι έχει μία πολύ φιλική προς το χρήστη διεπαφή (user interface) κάνουν το Ganache ένα πολύ βολικό εργαλείο για χρήση και ανάπτυξη αποκεντρωμένων εφαρμογών.

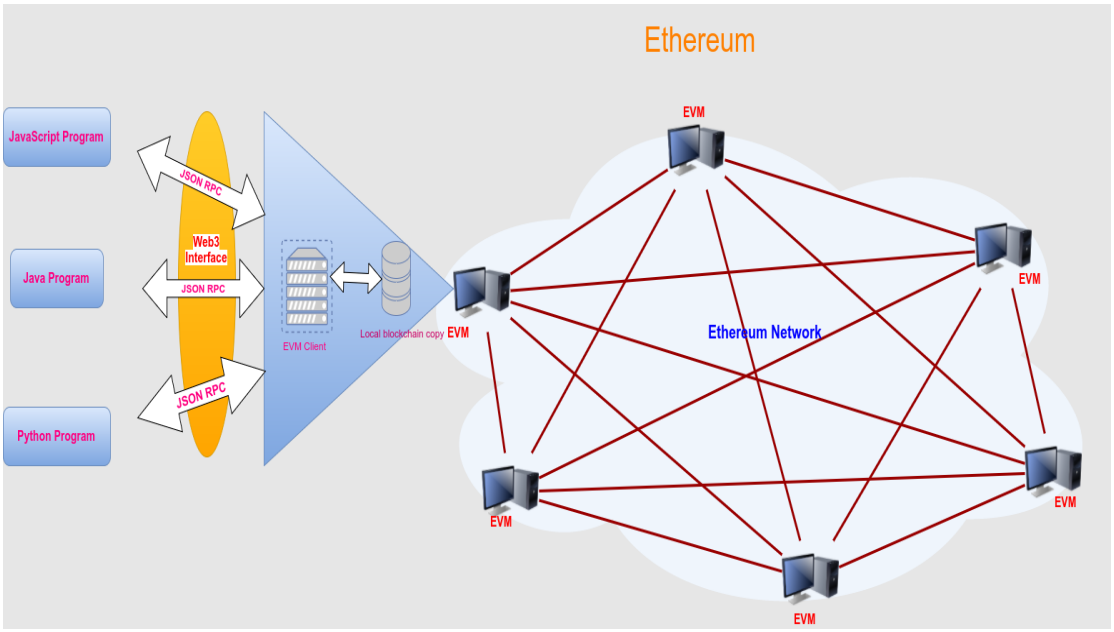
## 6.4 Python

Πέρα από τη solidity η οποία χρησιμοποιήθηκε για την ανάπτυξη του έξυπνου συμβολαίου, χρησιμοποιήθηκε και η Python για τη συγγραφή του κώδικα ο οποίος θα "τρέχει" τοπικά σε κάθε κόμβο. Η python [77] είναι μια διερμηνευόμενη (interpreted) γλώσσα προγραμματισμού υψηλού επιπέδου και δυναμικού τύπου. Υποστηρίζει τόσο αντικειμενοστραφή όσο και συναρτησιακό αλλά και διαδικαστικό προγραμματισμό [18]. Μπορεί να χρησιμοποιηθεί τόσο για τη δημιουργία σεναρίων εντολών (scripts) όσο και ως συνδετικός κρίκος πολλών επιμέρους προγραμμάτων (glue language) [77]. Υποστηρίζει τόσο αρθρώματα (modules) όσο και πακέτα (packages) [78]. Γενικά τα προγράμματα της python είναι συμπαγή και ευανάγνωστα. Διαθέτει πάρα πολλές βιβλιοθήκες πράγμα το οποίο κάνει τη ζωή των προγραμματιστών ευκολότερη. Στο πλαίσιο της παρούσας διπλωματικής χρησιμοποιήθηκε η έκδοση 3.7.3 της python.

## 6.5 Web3.py

Η web3.py [79] είναι μια βιβλιοθήκη της python που μας επιτρέπει να επικοινωνούμε με το Ethereum. Το API της προέρχεται από το Javascript API της web3.js.

Ας δούμε από την εικόνα 6.1 πως λειτουργεί η web3.py :




Εικόνα 6.1: Λειτουργία της web3.py [80]

Η web3.py επικοινωνεί με το Ethereum blockchain μέσω κλήσεων JSON RPC [80]. Όπως ήδη έχει εξηγηθεί στην ενότητα 3.8, στο Ethereum κάθε κόμβος έχει μια κόπια από τους κώδικες και τα δεδομένα του δικτύου. Μέσω της web3.py μπορούμε να κάνουμε αιτήματα (requests) σε έναν κόμβο του Ethereum (που δουλεύει για λογαρια-

σμό όλου του δικτύου). Έτσι μπορούμε να έχουμε πρόσβαση και να διαβάσουμε δεδομένα του Blockchain αλλά και να προσθέσουμε νέα δεδομένα σε αυτό. Τελικά η web3.py είναι μια πολύ σημαντική βιβλιοθήκη η οποία θα επιτρέψει στον κώδικα Python που έχουμε γράψει να καλεί συναρτήσεις του έξυπνου συμβολαίου και είτε να διαβάζει δεδομένα που χρειάζεται είτε να γράφει νέα δεδομένα στο blockchain.

## 6.6 Άλλες βιβλιοθήκες της Python που χρησιμοποιήθηκαν

**Numpy**: Η Numpy [81] είναι ένα θεμελιώδες πακέτο για επιστημονικούς υπολογισμούς που χρησιμοποιείται ευρέως. Στο πλαίσιο της συγκεκριμένης διπλωματικής χρησιμοποιήθηκε κατά κόρον στον κώδικα της Python μιας και προσφέρει εύκολο χειρισμό των N-διάστατων πινάκων και πολλές έτοιμες συναρτήσεις για το χειρισμό τους.

**Pandas**: Η Pandas [82] είναι μια επίσης πολυχρησιμοποιημένη βιβλιοθήκη που χρησιμοποιεί εργαλεία για δομές δεδομένων και ανάλυση δεδομένων. Μας επιτρέπει να χειριζόμαστε αριθμητικούς πίνακες και χρονοσειρές. Στην παρούσα διπλωματική χρησιμοποιήθηκε προκειμένου να μπορούν να περαστούν τα δεδομένα τα οποία λαμβάνει το πρόγραμμα μας ως είσοδο με τη μορφή csv αρχείου, σε πίνακες.

**Matplotlib.pyplot**: Η matplotlib.pyplot [83] είναι και αυτή μια από τις πιο γνωστές και χρήσιμες βιβλιοθήκες και παρέχει τη δυνατότητα για γραφικές παραστάσεις με τρόπο ανάλογο του Matlab. Χρησιμοποιήθηκε στο πλαίσιο αυτής της διπλωματικής για την δημιουργία των διαφόρων γραφικών παραστάσεων .

# Κεφάλαιο 7

## *Ανάλυση του κώδικα της διπλωματικής*

Στον παρόν κεφάλαιο επιχειρείται μια ανάλυση του κώδικα της python που "τρέχει" τοπικά σε κάθε κόμβο του δικτύου καθώς και του κώδικα του συμβολαίου που έχει αναπτυχθεί στο τοπικό blockchain.

## 7.1 Ανάλυση κώδικα έξυπνου συμβολαίου

Αρχικά να υπενθυμίσουμε ότι το έξυπνο συμβόλαιο αφού συλλέξει τις λύσεις $x_i(k+1)$ ενός προβλήματος βελτιστοποίησης που λύνει κάθε κόμβος, στη συνέχεια υπολογίζει με βάση τις τιμές που συνέλεξε, τις τιμές $y_i(k+1)$ καθώς και την ανανέωση του πολλαπλασιαστή Lagrange κάθε κόμβου $\lambda_i(k+1)$. Ουσιαστικά λύνει το ακόλουθο πρόβλημα βελτιστοποίησης:

$$min \sum_{i=1}^{N} \left[\lambda_i(k) \cdot (x_i(k+1) - y_i) + \frac{p}{2} \cdot (x_i(k+1) - y_i)^2\right])$$

$$s.to: \sum_{i=1}^{N} \mathrm{y_i} = \mathrm{P_d}$$

Και βρίσκε τα ακόλουθα :

$$\eta = \frac{\rho}{N} \cdot \left(P_d - \sum_{i=1}^{N} \left[x_i(k+1) + \frac{\lambda_i(k)}{\rho}\right]\right)$$

$$y_i(k+1) = \frac{\eta}{\rho} + \frac{\lambda_i(k)}{\rho} + x_i(k+1)$$

$$\lambda_i(k+1) = \lambda_i(k) + p \cdot \big(x_i(k+1) - \ y_i(k+1)\big)$$

Τέλος ελέγχει αν έχει επιτευχτεί σύγκλιση, δηλαδή αν ισχύει :

$|(x_i(k+1) - y_i(k+1)| < \varepsilon$ και $| - \rho \cdot (y_i(k+1) - y_i(k)| < \varepsilon$ για όλες τις γεννήτριες –κόμβους

Επίσης πρέπει να τονιστεί ότι η solidity σαν γλώσσα δεν υποστηρίζει προς το παρόν τους αριθμούς κινητής υποδιαστολής (floats). Οι κόμβοι στέλνουν τα δεδομένα τους στο έξυπνο συμβόλαιο πολλαπλασιασμένα επί 1000. Στη συνέχεια εκτελούνται οι πράξεις στο έξυπνο συμβόλαιο και τα αποτελέσματα που προκύπτουν (τα $y_i(k+1)$ και τα $\lambda_i(k+1)$) είναι πολλαπλασιασμένα επί 1000. Οπότε με αυτό τον τρόπο κρατάμε ακρίβεια τριών δεκαδικών ψηφίων. Προσπαθήσαμε να πολλαπλασιάσουμε και με μεγαλύτερα νούμερα π.χ. επί $10^5$ αλλά τότε η solidity έκανε υπερχείλιση (overflow). ηηηη

Παρουσιάζουμε λοιπόν αναλυτικά το έξυπνο συμβόλαιο :

```
pragma solidity ^0.5.0;
```

Αρχικά ορίζουμε την έκδοση του μεταγλωττιστή (compiler) που θα χρησιμοποιήσουμε.

```
uint public N;
int[1000][1000] public l;
int [1000][1000] public z;
int p;
int Pd;
int [1000] public h;
int[1000][1000] public x;
int[1000][1000] public y;
uint16 public k;
int epsilon;
int f_flag;
address[] public whitelist;
    mapping (address => bool) public waiting;
    bool public problemSolved;
    bool public init;
```

Στη συνέχεια ορίζουμε κάποιες μεταβλητές οι οποίες αποθηκεύονται στο storage του συμβολαίου. Δηλαδή η τιμές τους δεν "χάνονται" εκτός αν αλλάξουν μέσω του κώδικα. Το N είναι μια ακέραια μεταβλητή η οποία αναπαριστά το πλήθος των κόμβων που συμμετέχουν στο πρόβλημα Οικονομικής Κατανομής Φορτίου. Το l είναι ένας πίνακας που αποθηκεύει τον πολλαπλασιαστή Lagrange κάθε κόμβου σε κάθε επανάληψη $k$ (γενικά οι γραμμές αναπαριστούν τις επαναλήψεις του αλγορίθμου και οι στήλες την τιμή του πολλαπλασιαστή Lagrange κάθε κόμβου στην αντίστοιχη επανάληψη). Το z είναι ένας πίνακας που χρησιμοποιείται ώστε να κρατάμε σε κάθε επανάληψη $k$ το $\sum_{i=1}^{N}\left[x_i(k+1)+\frac{\lambda_i(k)}{\rho}\right]$ που απαιτείται για τον υπολογισμό του πολλαπλασιαστή Lagrange $\eta$ του προβλήματος βελτιστοποίησης που επιλύεται στο έξυπνο συμβόλαιο. Το p αποθηκεύει την παράμετρο $\rho$. Το Pd αποθηκεύει το φορτίο του συστήματος. Το h είναι ένας πίνακας που αποθηκεύει σε κάθε επανάληψη $k$ το πολλαπλασιαστή Lagrange $\eta$ του προβλήματος βελτιστοποίησης που λύνει το έξυπνο συμβόλαιο. Οι πίνακες x και y είναι οι λύσεις των προβλημάτων βελτιστοποίησης που λύνουν οι κόμβοι και το συμβόλαιο αντίστοιχα (η αποθήκευση στους πίνακες γίνεται με τη λογική που γίνεται η αποθ'ηκευση στοιχείων στον πίνακα l). Το k είναι η επανάληψη του αλγορίθμου. Το epsilon είναι το $\varepsilon$ σύγκλισης (γενικά επειδή το έξυπνο συμβόλαιο υπολογίζει τα $x_i$, $y_i$,$\lambda_i$ πολλαπλασιασμένα επί 1000 θα πρέπει και το $\varepsilon$ να ορίζεται επί 1000). Το whitelist είναι ένας πίνακας από διευθύνσεις όπου θα καταχωρούνται οι διευθύνσεις των κόμβων που συμμετέχουν στο πρόβλημα Οικονομικής Κατανομής φορτίου. Το waiting είναι ένα mapping (δηλαδή συσχετιστικός πίνακας) που αντιστοιχίζει μεταβλητές που είναι διευθύνσεις με μία λογική μεταβλητή. Εάν η μεταβλητή waiting μιας διεύθυνσης είναι αληθής (true) σημαίνει ότι αυτή διεύθυνση δεν έχει στείλει ακόμα δεδομένα και άρα την "περιμένουμε" να στείλει, ενώ όταν είναι ψευδής (false) σημαίνει ότι ήδη έχει στείλει τα δεδομένα της. Η Boolean μεταβλητή problemSolved είναι αληθής όταν το πρόβλημα Οικονομικής Κατανομής Φορτίου έχει λυθεί, ενώ η init είναι αληθής όταν βρισκόμαστε στη φάση αρχικοποίησης.

Γενικά επειδή η Εικονική Μηχανή του Ethereum είναι μηδενικής κατάστασης (zero state), αν οι ακέραιες μεταβλητές δεν αρχικοποιηθούν, τότε η Εικονική Μηχανή τις αρχικοποιεί αυτόματα ως μηδέν. Έτσι το k αρχικοποιείται στο μηδέν και ο πίνακας l αρχικοποιείται με μηδενικά. Οι πίνακες x και y θα αρχικοποιηθούν κατά τη φάση αρχικοποίησης.

Το f_flag είναι μια ακέραια μεταβλητή η οποία χρησιμοποιείται ώστε να μπορεί ο δημιουργός του συμβολαίου να ορίζει δεκαδικές τιμές στην παράμετρο $\rho$. Θα εξηγηθεί παρακάτω η λειτουργία της.

```
constructor (address[] memory _whitelist,uint _N ,int _p, int _Pd, int _epsilon, int _f_flag) public{
    p=_p;
    N=_N;
    Pd=_Pd*1000;
    whitelist=_whitelist;
    resetWaiting();
    problemSolved=false;
    init=true;
    k=0;
    epsilon = _epsilon;
    f_flag = _f_flag;
}
```

Αυτή είναι η συνάρτηση-κατασκευαστής (constructor). Όταν ένα συμβόλαιο δημιουργείται, η συνάρτηση-κατασκευαστής εκτελείται μόνο μια φορά. Αφού εκτελεστεί το έξυπνο συμβόλαιο, ο τελικός κώδικας του συμβολαίου αναπτύσσεται στο blockchain χωρίς να περιλαμβάνει τον κώδικα της συνάρτησης-κατασκευαστή [73]. Μέσω αυτής της συνάρτησης αρχικοποιούνται κάποιες μεταβλητές. Η μεταβλητή problemSolved γίνεται false (δεν έχει λυθεί το πρόβλημα οικονομικής κατανομής φορτίου), η μεταβλητή init γίνεται true (είναι η φάση αρχικοποίησης). Επίσης ο δημιουργός του συμβολαίου ορίζει κάποιες άλλες παραμέτρους. Καταρχάς ορίζει την παράμετρο $\rho$ , τον αριθμό των κόμβων N, το φορτίο του συστήματος Pd, την λίστα των διευθύνσεων που μπορούν να συμμετέχουν στο πρόβλημα οικονομικής κατανομής φορτίου (ιδιωτικό blockchain), το $\varepsilon$ σύγκλισης, αλλά και το f_flag. Το f_flag είναι μηδέν όταν ο αριθμός $\rho$ είναι ακέραιος και 1 όταν έχει ένα δεκαδικό.

```
function submitValue(int value,uint i, uint16 iteration ) public{
    require(!problemSolved);
    require(iteration==k);
     if(waiting[msg.sender])
     {
          if(msg.sender==whitelist[i])
             {
             x[k+1][i]=value;
             waiting[msg.sender]=false;
             }
      else{revert();}
     }
else{revert();}

if(!stillWaiting())
   {
        if (f_flag == 0) updateY();
        else updateY_float();
        resetWaiting();
        k++;

   }

}
```

Η συνάρτηση submitInitalValue καλείται από τους κόμβους που συμμετέχουν στο πρόβλημα της Οικονομικής Κατανομής Φορτίου έτσι ώστε να ενημερώσουν το έξυπνο συμβόλαιο με ποια τιμή θα ξεκινήσουν να λύνουν το πρόβλημα βελτιστοποίησης τους. Ελέγχεται καταρχάς αν βρισκόμαστε σε φάση αρχικοποίησης και αν έχει λυθεί το πρόβλημα Οικονομικής Κατανομής Φορτίου. Στη συνέχεια κάθε κόμβος στέλνει την τιμή $x_i(0)$ με την οποία θα ξεκινήσει να λύνει το πρόβλημα βελτιστοποίησης του, καθώς και έναν ακέραιο αριθμό που δηλώνει σε ποια θέση στη whitelist βρίσκεται. Το έξυπνο συμβόλαιο αφού ελέγξει ότι πράγματι περιμένουμε αυτόν τον κόμβο να στείλει δεδομένα και ότι η διεύθυνση που έχει υπογράψει τη συναλλαγή (msg.sender) βρίσκεται στη θέση i στη whitelist τότε θέτει το $y_i(0)$ ίσο με $x_i(0)$ και κάνει τη μεταβλητή waiting αυτής της διεύθυνσης ίση με false (ώστε να μην μπορεί να ξαναστείλει δεδομένα σε αυτή συνάρτηση). Ουσιαστικά το πρόβλημα βελτιστοποίησης που επιλύεται στο έξυπνο συμβόλαιο ξεκινά με τις ίδιες τιμές που ξεκινά το πρόβλημα βελτιστοποίησης που επιλύεται τοπικά σε κάθε κόμβο. Τέλος το συμβόλαιο ελέγχει αν η συνάρτηση stillWaiting (είναι μια συνάρτηση που επιστρέφει false όταν το waiting είναι False για όλες τις διευθύνσεις, δηλαδή όταν όλοι οι κόμβοι έχουν στείλει δεδομένα και δεν περιμένουμε κανέναν ακόμα) επιστρέφει false που θα σημάνει τη λήξη της φάσης αρχικοποίησης (init=false) και θα καλέσει και τη resetWaiting() που είναι μια συνάρτηση όπου θα θέσει το waiting κάθε κόμβου ίσο με True ώστε να τον περιμένουμε να στείλει δεδομένα για την επομένη φάση του αλγορίθμου.

```
function submitValue(int value,uint i, uint16 iteration ) public{
    require(!problemSolved);
    require(iteration==k);
     if(waiting[msg.sender])
     {

       if(msg.sender==whitelist[i])
          {
             x[k+1][i]=value;
             waiting[msg.sender]=false;
          }
       else { revert();}
     }
else{revert();}

    if(!stillWaiting()){

        if (f_flag == 0)
        {
            updateY();

        }
        else {updateY_float();}
    resetWaiting();
    k++;
   }

}
```

Η συνάρτηση submitValue λειτουργεί περίπου σαν την submitInitialValue. Κάθε κόμβος σε κάθε επανάληψη $k$ στέλνει το $x_i(k+1)$ που έχει υπολογίσει, τη θέση του i στη whitelist αλλά και την επανάληψη στην οποία βρίσκεται το πρόβλημα βελτιστοποίησης το οποίο λύνει τοπικά. Αυτό γίνεται για να ταυτοποιηθεί ότι το πρόβλημα βελτιστοποίησης που λύνουν οι κόμβοι τοπικά αλλά και το πρόβλημα βελτιστοποίησης που επιλύεται στο blockchain είναι στην ίδια επανάληψη. Αφού στείλουν όλοι οι κόμβοι τις λύσεις τους και αρά δεν περιμένουμε κάποιον, ελέγχεται αν η f_flag είναι 0 ώστε να εκτελεστεί η συνάρτηση updateY ή αν είναι 1 ώστε να εκτελεστεί η updateY_float. Μετά την εκτέλεσης είτε της updateY είτε της updateY_ float (εκεί υλοποιούνται οι υπολογισμοί των $y_i(k+1)$ και $\lambda_i(k+1)$ και ελέγχεται αν έχει επέλθει σύγκλιση) θέτουμε το waiting κάθε κόμβου ίσο με true ώστε να τον περιμένουμε για την επόμενη επανάληψη (αν δεν έχει επιτευχθεί σύγκλιση) .

```
function updateY() public {
    for(uint i=0; i<N; i++){
        // compute y[k+1][i], l[k+1][i] if p int

        if(i==0) z[k][i]=x[k+1][i]+(l[k][i])/p;
        else z[k][i]=z[k][i-1]+x[k+1][i]+(l[k][i])/p;

    }

     h[k]=(p*((Pd)-z[k][N-1]))/int(N);

    for(uint i; i<N; i++){

      y[k+1][i]=(h[k]/p)+x[k+1][i]+(l[k][i])/p;
      l[k+1][i]=l[k][i]+p*(x[k+1][i]-y[k+1][i]);
    }

    //check for convergence


    for(uint i=0; i<N; i++){
        if(abs(x[k+1][i],y[k+1][i]) >epsilon) return;
        if((abs(y[k+1][i],y[k][i]))*p>epsilon) return;
    }
    problemSolved = true;
}
```

Αν η f_flag είναι ίση με 0 (άρα $\rho$:ακέραιος) τότε αρχικά υπολογίζεται το $\sum_{i=1}^{N}\left[x_i(k+1)+\frac{\lambda_i(k)}{\rho}\right]$ ώστε να υπολογιστεί το η και στη συνέχεια υπολογίζεται το $y_i(k+1)$ και το $\lambda_i(k+1)$ για κάθε κόμβο και ελέγχουμε για σύγκλιση. Αν πράγματι ο αλγόριθμος έχει συγκλίνει τότε η μεταβλητή problemSolved γίνεται true και το πρόβλημα οικονομικής κατανομής φορτίου έχει λυθεί. Αν δεν έχει επιτευχθεί σύγκλιση τότε πηγαίνουμε στην επόμενη επανάληψη αφού πρώτα όπως είπαμε και παραπάνω το waiting για κάθε κόμβο γίνει true.

```
function updateY_float() public {
       // compute y[k+1][i], l[k+1][i] if p float

    for(uint i=0; i<N; i++){
        if(i==0) z[k][i]=x[k+1][i]+(l[k][i])*p;
        else z[k][i]=z[k][i-1]+x[k+1][i]+(l[k][i])*p;

    }

     h[k]=(((Pd)-z[k][N-1]))/(p*int(N));

    for(uint i; i<N; i++){

      y[k+1][i]=(h[k]*p)+x[k+1][i]+(l[k][i])*p;
      l[k+1][i]=l[k][i]+(x[k+1][i]-y[k+1][i])/p;
    }


    //check for convergence

    for(uint i=0; i<N; i++){
        if(abs(x[k+1][i],y[k+1][i]) >epsilon) return;
        if(((abs(y[k+1][i],y[k][i]))/p)>epsilon) return;
    }
    problemSolved = true;

}
```

Αν το $\rho$ είναι δεκαδικός μικρότερος της μονάδας τότε καλείται η συνάρτηση updateY_float. Οι πράξεις διαφοροποιούνται ως εξής: όπου το $\rho$ είναι στο παρονομαστή τότε αντί για διαίρεση δια ρ κάνουμε πολλαπλασιασμό επί 1/ $\rho$ και όπου το $\rho$ είναι στον αριθμητή κάνουμε διαίρεση διά ρ. Δηλαδή έστω ότι θέλουμε το $\rho$ να πάρει τη τιμή 0,1. Τότε όπου έχουμε διαίρεση με $\rho$, αντί για διαίρεση δια 0,1 κάνουμε πολλαπλασιασμό επί 10 και όπου έχουμε πολλαπλασιασμό με $\rho$ τότε κάνουμε διαίρεση δια 10. Αυτό το τέχνασμα γίνεται διότι η solidity δεν υποστηρίζει δεκαδικούς. Βέβαια και πάλι αυτό το τέχνασμα δεν μπορεί να εφαρμοστεί για όλες τις μη ακέραιες τιμές του ρ. Μπορεί να εφαρμοστεί μόνο για τιμές του $\rho$ κάτω της 1 και μόνο για περιπτώσεις όπου το αποτέλεσμα του 1/ $\rho$ είναι ακέραιος αριθμός. Έτσι λοιπόν ο δημιουργός του συμβολαίου όταν θέλει να ορίσει ρ δεκαδικό κάτω της μονάδας θα πρέπει να ορίσει την f_float=1 και στη παράμετρο p να περάσει ως τιμή το 1/ρ.

## 7.2 Ανάλυση κώδικα Python

Όπως έχουμε αναφέρει κάθε κόμβος τοπικά επιλύει ένα πρόβλημα βελτιστοποίησης της ακόλουθης μορφής:

$$min\, C_i(x_i) = \min\left(f_i(x_i) + \lambda_i(k)\cdot(x_i - y_i(k)) + \frac{p}{2}\cdot(x_i - y_i(k))^2\right)$$

$$\text{s.to}: x_i^{min} \le x_i \le x_i^{max}$$

Κάθε κόμβος βρίσκει λοιπόν μια λύση:

$$x_i(k+1) = min\left\{max\left\{\frac{-b_i - \lambda_i(k) + p\cdot y_i(k)}{2a_i + p}, x_i^{min}\right\}, x_i^{max}\right\}$$

Αυτή τη λύση του τη στέλνει στο έξυπνο συμβόλαιο Aggregator που υπάρχει στο blockchain .

Στη συνέχεια αναλύονται τα βασικά σημεία του κώδικα της python:

```
web3=Web3(Web3.HTTPProvider('HTTP://127.0.0.1:7545'))
```

Αρχικά πρέπει να επιτύχουμε σύνδεση με το Ganache (το blockchain). Ο τοπικός διακομιστής (local host) του Ganache είναι: 'HTTP://127.0.0.1:7545'. Χρησιμοποιούμε λοιπόν την κλάση Providers της Web3 και συνδεόμαστε με το ganache μέσω του HTTP πρωτοκόλλου (Οι providers παίρνουν JSON-RPC αιτήματα και επιστρέφουν την απάντηση. Αυτό γίνεται υποβάλλοντας το αίτημα σε διακομιστή HTTP ή IPC υποδοχής [79]).

```
contract_address="0x3Db6e2407F41EaE4D6f08e63e0d454F21121AC7a"
abi =json.loads('abi')
contract=web3.eth.contract(address=contract_address , abi=abi)
```

Στη συνέχεια προκειμένου ο κώδικας της python να μπορεί να επικοινωνεί με το έξυπνο συμβόλαιο θα πρέπει να μπορούμε με κάποιο τρόπο να αναπαραστήσουμε το έξυπνο συμβόλαιο μέσα στον κώδικα μας. Για να το επιτύχουμε αυτό χρειαζόμαστε δύο πράγματα [79]:

- Το ABI (Abstract Binary Interface) του συμβολαίου, που είναι ένα json αρχείο το οποίο ορίζει τις λειτουργιές του συμβολαίου.
- Την διεύθυνση του συμβολαίου στο Ethereum

Το ABI του συμβολαίου υπάρχει στο remix και η διεύθυνση του συμβολαίου που δημιουργείται όταν το συμβόλαιο αναπτυχθεί (deploy) στο Ganache μπορεί να βρεθεί είτε από το Ganache είτε από το Remix.

Όπως βλέπουμε κρατάμε στη μεταβλητή contract_address την διεύθυνση του συμβολαίου στο Ganache και στη μεταβλητή abi αποθηκεύουμε το ABI του συμβολαίου. Επειδή το ABI του συμβολαίου είναι αρκετά μακροσκελές, δεν παρουσιάζεται στο παραπάνω κομμάτι κώδικα (κανονικά το ABI μπαίνει μέσα στην παρένθεση τη δεύτερης εντολής του παραπάνω κώδικα εντός των ' ').

Τέλος η web3 συνδυάζοντας το ABI του συμβολαίου με τη διεύθυνση του, δημιουργεί μία αναπαράσταση του συμβολαίου η οποία αποθηκεύεται στη μεταβλητή contract.

```
addresses=web3.eth.accounts
```

Στη συνέχεια αποθηκεύουμε στον πίνακα addresses τις διευθύνσεις όλων των λογαριασμών του Ganache.

```
problemSolved = False
mul = 1000; #because solidity doed not support floats
```

Ορίζουμε μια μεταβλητή problemSolved η οποία αρχικοποιείται σε False και ουσιαστικά όταν γίνει True το πρόβλημα Οικονομικής Κατανομής Φορτίου θα έχει λυθεί. Ορίζουμε επίσης το mul =1000 με το οποίο θα πολλαπλασιάζονται οι λύσεις που βρίσκουν οι κόμβοι πριν τις στείλουν στο Ganache. Αυτό το κάνουμε διότι όπως εξηγήσαμε και πριν η solidity δεν υποστηρίζει δεκαδικούς αριθμούς. Έτσι λοιπόν προκειμένου να περιορίσουμε την απώλεια σε ακρίβεια, οι κόμβοι στέλνουν τα δεδομένα πολλαπλασιασμένα επί 1000.

```
#init ( send the initial values x[0][i])
k=0
for i in range(N):
 web3.eth.defaultAccount=addresses[i]
 tx_hash=contract.functions.submitInitialValue(int(x[k,i]*mul) ,i).transact()
 web3.eth.waitForTransactionReceipt(tx_hash,timeout=120)
```

Στη συνέχει όλοι κόμβοι στέλνουν στο έξυπνο συμβόλαιο τις αρχικές τιμές $x_i(0)$ με τις οποίες θα αρχίσουν να λύνουν το πρόβλημα βελτιστοποίησης. Για να γίνει αυτό ο κάθε κόμβος του Ganache δημιουργεί μία συναλλαγή και την υπογράφει με το ιδιωτικό του κλειδί. Αυτό ουσιαστικά κάνει το defaultAccount [79].

Στη συνέχεια οι κόμβοι καλούν τη συνάρτηση submitInitalValue του έξυπνου συμβολαίου και στέλνουν το $x_i(0)$ πολλαπλασιασμένο επί 1000 καθώς και τη θέση τους στην λίστα whitelist. Έτσι δημιουργείται ένα hash της συναλλαγής που αποθηκεύεται στη μεταβλητή tx_hash. Τέλος ο κάθε κόμβος βεβαιώνεται ότι η συναλλαγή του έχει μπει σε block λαμβάνοντας την απόδειξη της συναλλαγής. Επίσης όπως βλέπουμε έχει οριστεί και ένα χρονικό περιθώριο 120 δευτερολέπτων μέχρι να μπει η συναλλαγή στο block.

```
l[k][i]=contract.functions.l(k,i).call()
y[k][i]=contract.functions.y(k,i).call()
```

Αφού έγινε η φάση αρχικοποίησης, περνάμε στο βασικό κομμάτι του αλγορίθμου. Αρχικά πρέπει να αναφέρουμε ότι ο μεταγλωττιστής της solidity δημιουργεί getter συναρτήσεις για όλες τις δημόσιες μεταβλητές κατάστασης π.χ. για μία δημόσια ακέραια μεταβλητή κατάστασης δημιουργεί μια συνάρτηση που επιστρέφει ακέραιο [73]. Έτσι κάθε κόμβος διαβάζει από το συμβόλαιο την τιμή $y_i(k)$ καθώς και την $\lambda_i(k)$ (τα y και l στο έξυπνο συμβόλαιο είναι δημόσιες μεταβλητές κατάστασης και άρα δημιουργούν getter συναρτήσεις) .

```
l[k][i] = l[k][i]/(mul)
y[k][i] = y[k][i]/(mul)
```

Στη συνέχεια διαιρούν τις τιμές που διάβασαν δια 1000 μιας και η solidity παράγει αυτές τις μεταβλητές πολλαπλασιασμένες επί 1000. Θεωρούμε ότι οι πολλαπλασιαστές Lagrange αρχικά είναι μηδέν όποτε στην αρχή της πρώτη επανάληψης δεν δημιουργείται κάποιο λάθος. Μόλις η πρώτη επανάληψη τελειώσει οι τιμές των $\lambda_i(k+1)$ έχουν υπολογιστεί επί 1000.

```
x[k+1,i]=min(max((p*y[k,i]-l[k][i]-b[i])/(2*a[i]+p),x_min[i]),x_max[i])
web3.eth.defaultAccount=addresses[i]
tx_hash=contract.functions.submitValue(int(x[k+1,i]*mul) ,i,k).transact()
web3.eth.waitForTransactionReceipt(tx_hash,timeout=120)
```

Αφού οι κόμβοι διαβάσουν από το έξυπνο συμβόλαιο τις μεταβλητές $y_i(k)$ και $\lambda_i(k)$, βρίσκουν το $x_i(k+1)$ και το στέλνουν στο έξυπνο συμβόλαιο με τη λογική που είδαμε πριν.

```
problemSolved = contract.functions.problemSolved().call()
```

Τέλος διαβάζουν τη δημόσια Boolean μεταβλητή κατάστασης problemSolved από το έξυπνο συμβόλαιο και αν είναι Αληθής, το πρόβλημα οικονομική κατανομής φορτίου έχει τελειώσει αλλιώς συνεχίζουν στην επόμενη επανάληψη.

Αυτός είναι ο βασικός κώδικας της python. Πέρα από αυτό το κομμάτι, ο κώδικας της python περιλαμβάνει το κομμάτι των γραφικών παραστάσεων που θα παρουσιαστούν στο επόμενο κεφάλαιο καθώς και το "διάβασμα" του αρχείου εισόδου (csv). Σκόπιμο κρίνεται να εξηγηθεί πως μετά το τέλος του αλγορίθμου βρίσκουμε την κατανάλωση gas κάθε κόμβου. Όπως εξηγήσαμε προκειμένου η Εικονική Μηχανή του Ethereum να μπορεί να εκτελεί τις συναλλαγές, θα πρέπει ο κάθε κόμβος να πληρώνει κάθε υπολογιστικό βήμα με κάποιο ποσό χρημάτων (gasprice). Είναι λοιπόν σημαντικό να γνωρίζουμε την κατανάλωση gas κάθε κόμβου ώστε να γνωρίζουμε την οικονομική του επιβάρυνση κατά την εκτέλεση του αλγορίθμου.

```
for i in range (N):
    balances[i]=web3.eth.getBalance(addresses[i])
totalGas=(-balances+100.e+18)/gasprice
```

Καταρχάς με την εντολή getBalance βρίσκουμε το υπόλοιπο κάθε κόμβου μετά το πέρας του αλγορίθμου (σε wei). Έτσι λοιπόν γνωρίζοντας ότι όλοι οι λογαριασμοί έχουν αρχικά 100 ether ($100 \cdot 10^{18}$ wei) και με γνωστό το gas price, το συνολικό gas που κατανάλωσε κάθε κόμβος θα είναι οι λόγος των wei που ξόδεψε προς το gas price (wei/gas).

```
spentEther=(-balances+100.e+18)/(1.e+18)
maxGas=np.max(totalGas)
minGas=np.min(totalGas)
averageGas=np.average(totalGas)
averageEther=np.average(spentEther)
```

Τέλος βρίσκουμε τα ether που ξόδεψε κάθε κόμβος καθώς και τη μέση κατανάλωση ether, αλλά και την μέγιστη, την ελάχιστη και τη μέση τιμή της κατανάλωσης gas.

# Κεφάλαιο 8

## *Σενάρια διπλωματικής*

## 8.1 Βασικό σενάριο διπλωματικής- Σύστημα 6 γεννητριών

Όπως περιγράφεται και στο [72] υποθέτουμε ότι το σύστημά μας αποτελείται από έξι (6) γεννήτριες και οι παράμετροι των γεννητριών δίνονται από τον πίνακα 8.1. Το συνολικό φορτίο του συστήματος είναι $P_d = 150\ kW$.

| Κόμβος | $\alpha_i$ | $b_i$ | $c_i$ | $x_i^{min}$(kW) | $x_i^{max}$(kW) |
|---|---|---|---|---|---|
| 1 | 0.08 | 2 | 5 | 20 | 60 |
| 2 | 0.07 | 3.5 | 7 | 10 | 55 |
| 3 | 0.058 | 2.5 | 6.4 | 10 | 40 |
| 4 | 0.065 | 4 | 9.2 | 5 | 45 |
| 5 | 0.06 | 3 | 8.3 | 5 | 20 |
| 6 | 0.038 | 3.6 | 5.8 | 10 | 35 |

Πίνακας 8.1: Παράμετροι των 6 γεννητριών του συστήματος

Για τις επαναλήψεις (5.24)-(5.26) οι αρχικές τιμές δίνονται :

$$\boldsymbol{x}(0) = [35,25,30,28,12,20]^T$$

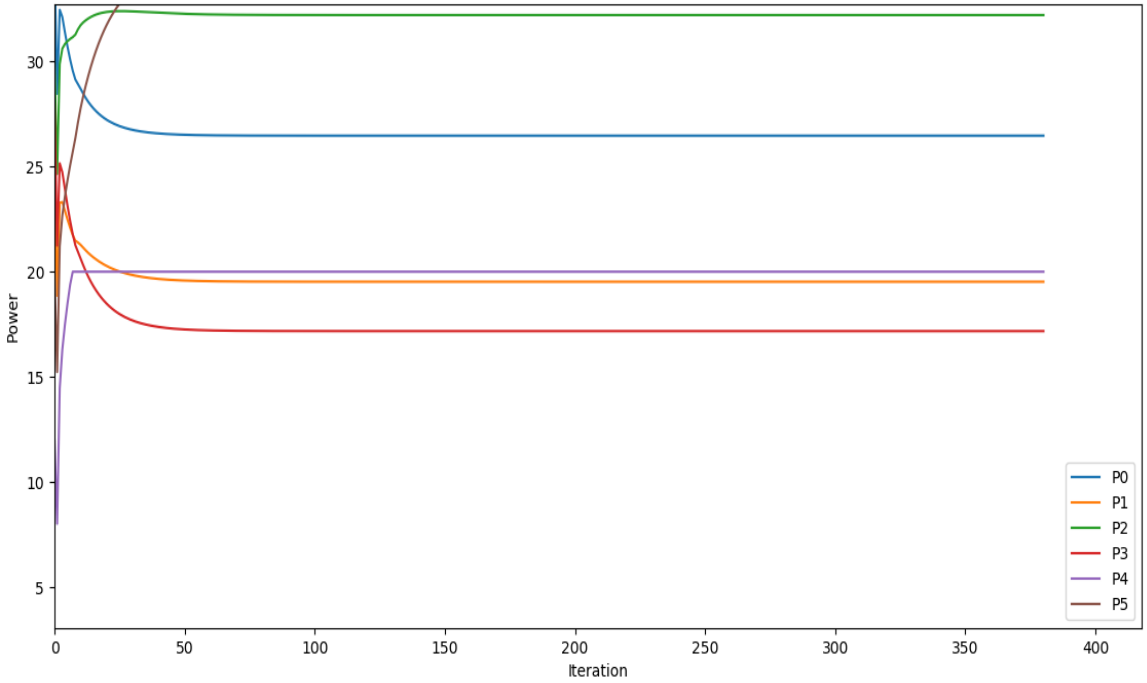
$$\boldsymbol{\lambda}(0) = 0$$

$$\boldsymbol{y}(0) = \boldsymbol{x}(0)$$

ενώ η τιμή της παραμέτρου $\rho$ επιλέγεται ως $\rho = 1$.

## 8.2 Επίλυση του βασικού σεναρίου με χρήση του κεντρικού αλγορίθμου

Αρχικά λύνουμε το πρόβλημα με χρήση του κεντρικού αλγορίθμου. Ορίζουμε ως όριο σύγκλισης $\varepsilon = 10^{-14}$ και υλοποιώντας τον αλγόριθμο έχουμε σύγκλιση μετά από 381 επαναλήψεις. Στο διάγραμμα 8.1 που ακολουθεί παρατηρούμε ότι η ισχύς εξόδου της κάθε γεννήτριας συγκλίνει στην βέλτιστη τιμή της που δίνεται στο πίνακα 8.2 :

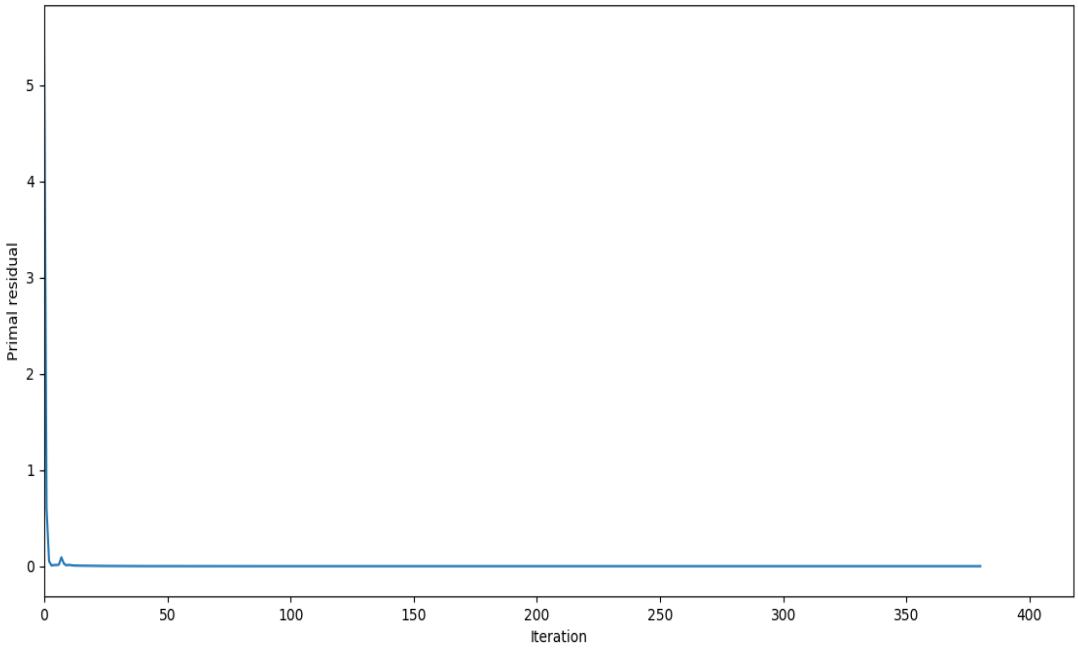



Διάγραμμα 8.1: Η εξέλιξη της ισχύος εξόδου κάθε γεννήτριας κατά τη διάρκεια του κεντρικού αλγορίθμου (για σύστημα 6 γεννητριών)

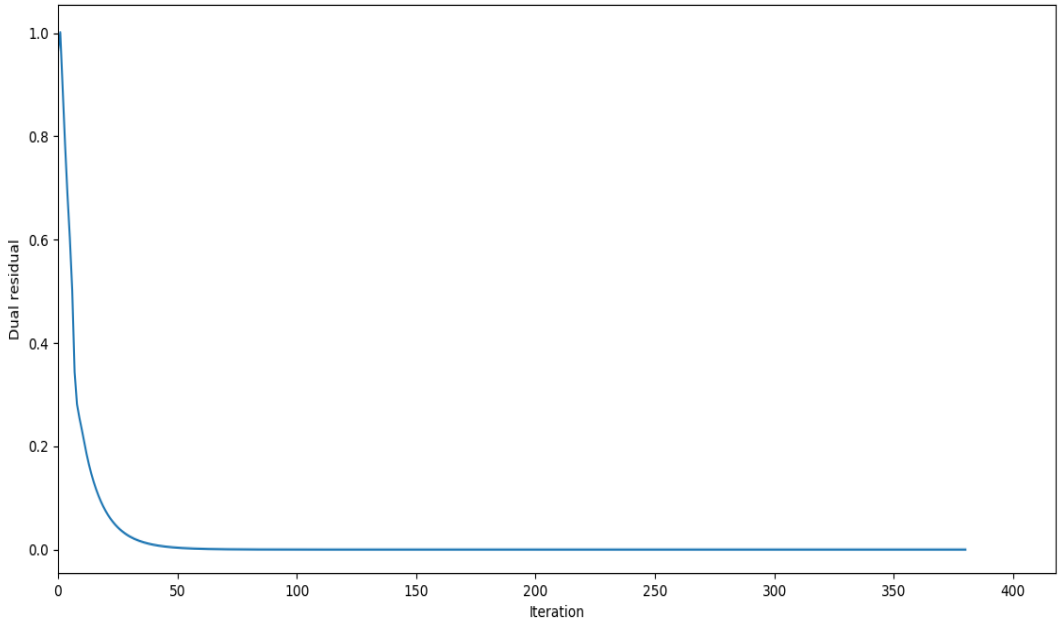
| Κόμβος | P(kW) |
|---|---|
| 1 | 26.459 |
| 2 | 19.525 |
| 3 | 32.185 |
| 4 | 17.180 |
| 5 | 20 |
| 6 | 34.651 |

Πίνακας 8.2: Ισχύς εξόδου της κάθε γεννήτριας μετά το πέρας του κεντρικού αλγορίθμου (για σύστημα 6 γεννητριών)

Στα διαγράμματα 8.2 και 8.3 παρουσιάζεται η εξέλιξη της primal και dual residual της γεννήτριας 1. Όπως βλέπουμε και οι δυο μεταβλητές συγκλίνουν στο μηδέν καθώς προχωράει ο αλγόριθμος, το οποίο δείχνει ότι η ισχύς εξόδου της γεννήτριας 1 μπορεί να συγκλίνει στη βέλτιστη τιμή της. Το ίδιο ισχύει και για τις άλλες γεννήτριες απλά ενδεικτικά επιλέξαμε να παρουσιάσουμε τη σύγκλιση της primal και dual residual της γεννήτριας 1.



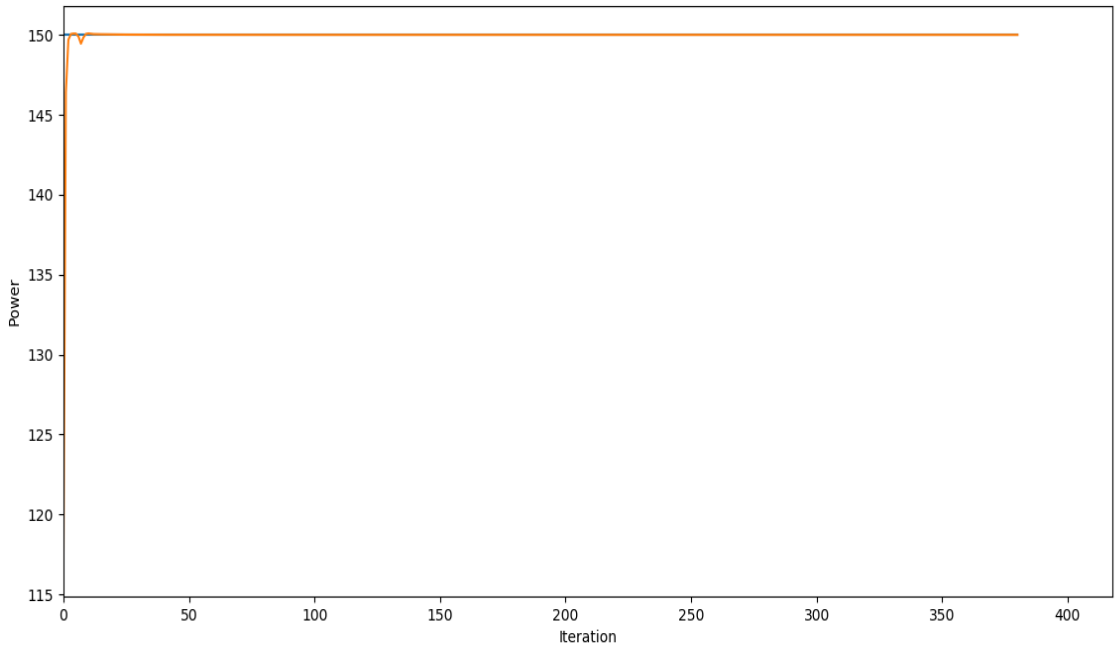

Διάγραμμα 8.2: Η εξέλιξη της primal residual της πρώτης γεννήτριας κατά τη διάρκεια του κεντρικού αλγορίθμου (για σύστημα 6 γεννητριών)



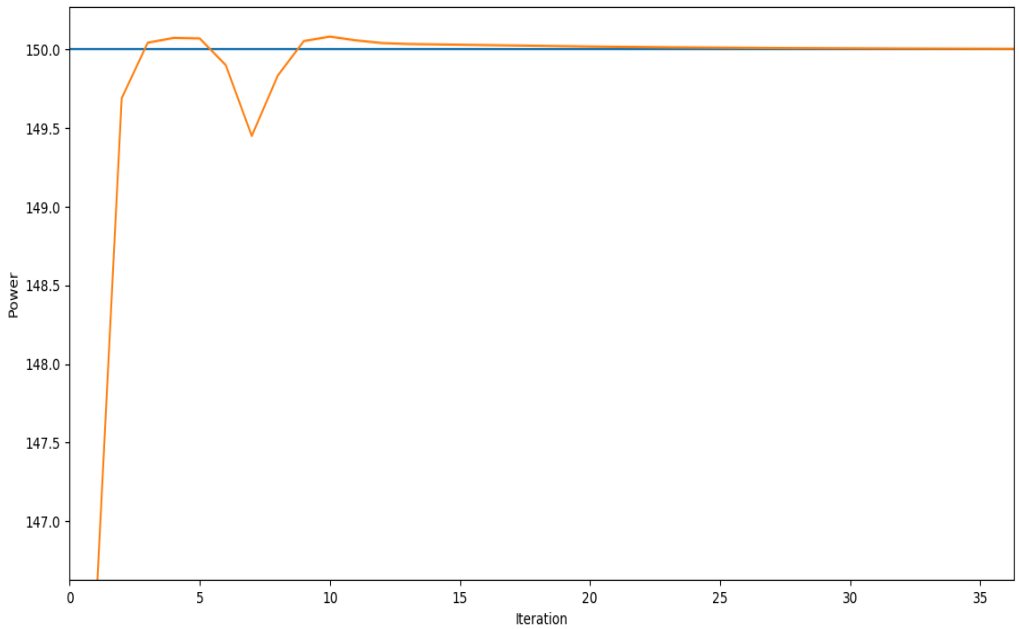

Διάγραμμα 8.3: Η εξέλιξη της dual residual της πρώτης γεννήτριας κατά τη διάρκεια του κεντρικού αλγορίθμου (για σύστημα 6 γεννητριών)

Τέλος στο διάγραμμα 8.4 που ακολουθεί παρατηρούμε την ισορροπία μεταξύ της παραγωγής και της ζήτησης κατά τη διάρκεια του αλγορίθμου. Συγκεκριμένα από το

διάγραμμα 8.5 που είναι μεγέθυνση του διαγράμματος 8.4 βλέπουμε πως η ισορροπία παραγωγής-ζήτησης επιτυγχάνεται μετά τις πρώτες 16 επαναλήψεις του αλγορίθμου.



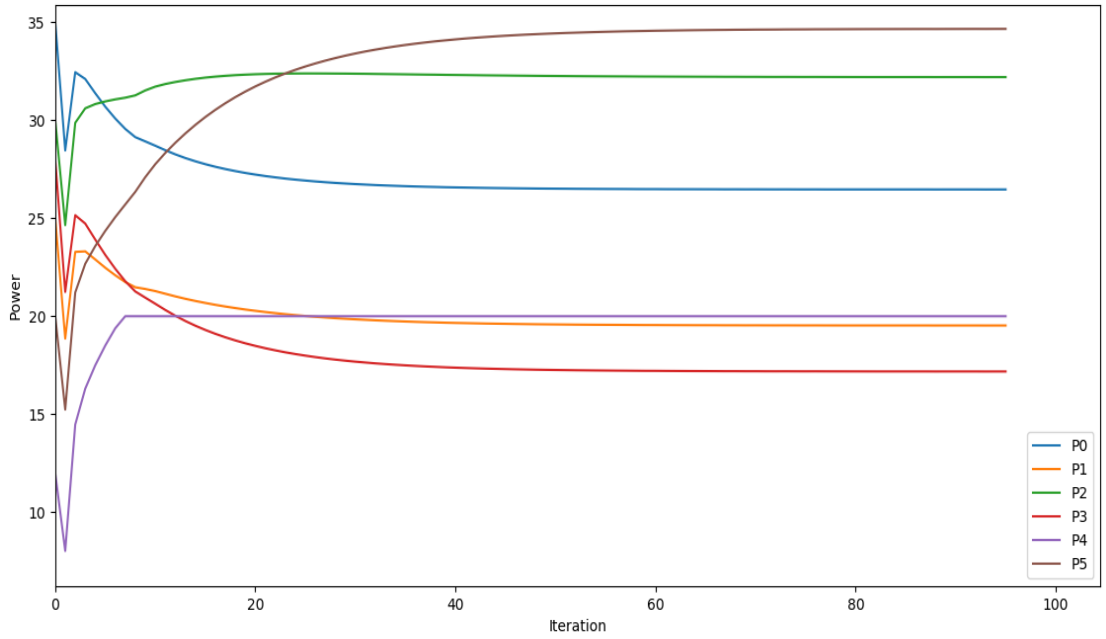

Διάγραμμα 8.4: Η εξέλιξη της ισορροπίας μεταξύ παραγωγής και ζήτησης κατά τη διάρκεια εκτέλεσης του κεντρικού αλγορίθμου (για σύστημα 6 γεννητριών)




Διάγραμμα 8.5: Μεγέθυνση του διαγράμματος 8.4

## 8.3 Επίλυση του βασικού σεναρίου με χρήση του αποκεντρωμένου αλγορίθμου

Επιλύουμε το παραπάνω πρόβλημα με τις έξι γεννήτριες, αλλά αυτή τη φορά χρησιμοποιούμε τον αποκεντρωμένο αλγόριθμο. Εκτελούμε τον αποκεντρωμένο αλγόριθμο (σχετικός κώδικας δίνεται στο παράρτημα) και το πρόβλημα Οικονομικής Κατανομής Φορτίου λύνεται μετά από 96. Η σύγκλιση επιτυγχάνεται σε λιγότερες επαναλήψεις σχέση μιας και στον αποκεντρωμένο αλγόριθμο κρατάμε ακρίβεια 3 δεκαδικών (όπως εξηγήθηκε και στο 7.1) ενώ στον κεντρικό κρατάμε ακρίβεια 14 δεκαδικών. Στο διάγραμμα 8.6 που ακολουθεί βλέπουμε την εξέλιξη της ισχύος κάθε γεννήτριας καθώς εκτελείται ο αλγόριθμος ενώ στον πινάκα 8.3 βλέπουμε την ενεργό ισχύ εξόδου κάθε γεννήτριας.

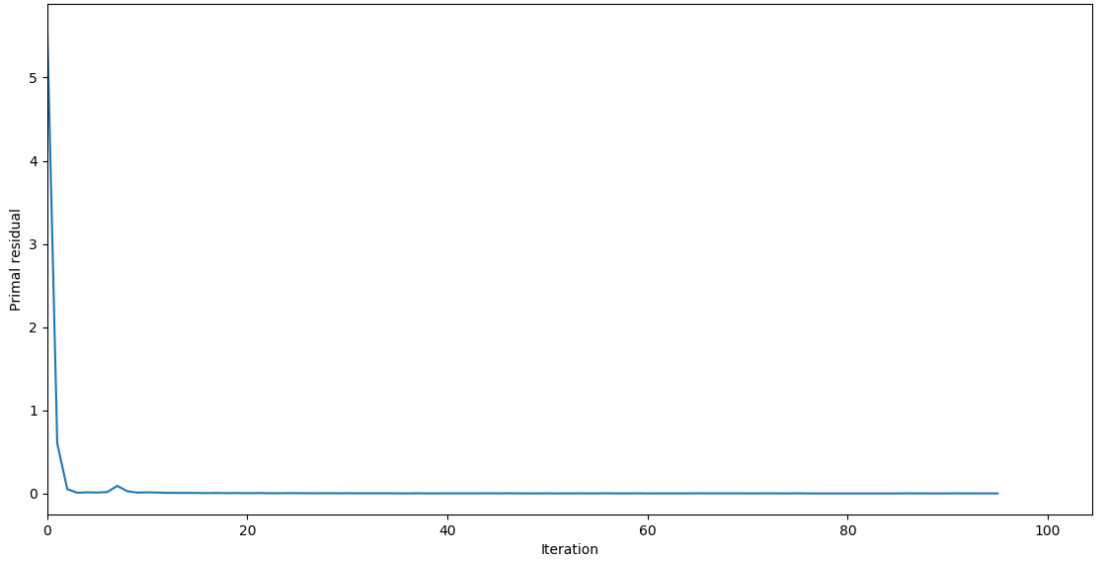



Διάγραμμα 8.6: Η εξέλιξη της ενεργού ισχύος εξόδου κάθε γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών)

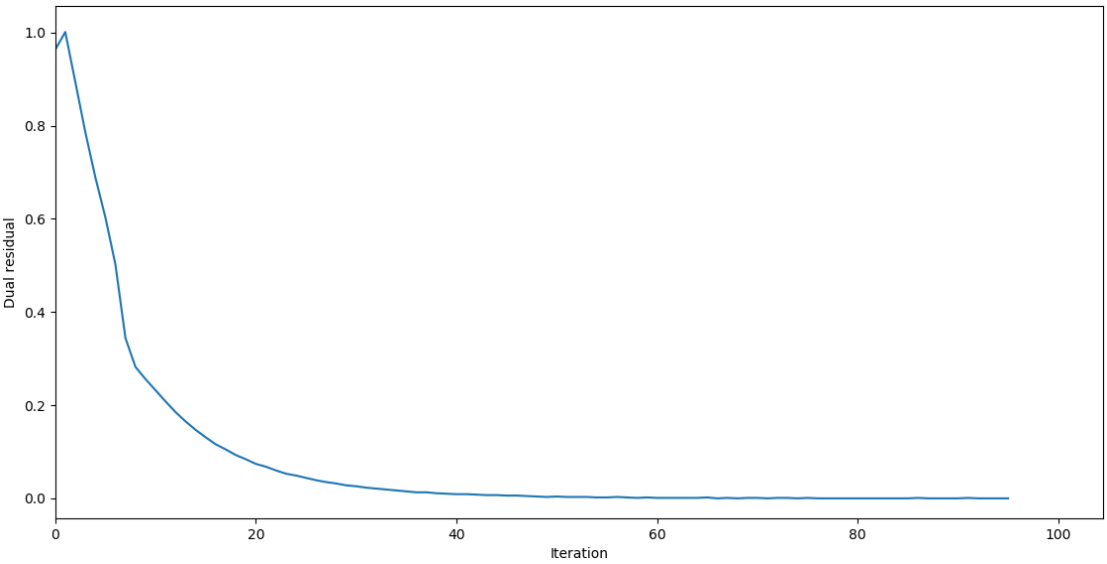
| Κόμβος | P(kW) |
|---|---|
| 1 | 26.460 |
| 2 | 19.526 |
| 3 | 32.187 |
| 4 | 17.182 |
| 5 | 20 |
| 6 | 34.645 |

Πίνακας 8.3: Ισχύς εξόδου της κάθε γεννήτριας μετά το πέρας του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών)

Στο διαγράμματα 8.7 και 8.8 παρουσιάζεται η εξέλιξη της primal και dual residual της γεννήτριας 1. Όπως βλέπουμε και οι δυο μεταβλητές συγκλίνουν στο μηδέν καθώς προχωράει ο αλγόριθμος, το οποίο δείχνει ότι η ισχύς εξόδου της γεννήτριας 1 μπορεί να συγκλίνει στη βέλτιστη τιμή της. Το ίδιο ισχύει και για τις άλλες γεννήτριες απλά ενδεικτικά επιλέξαμε να παρουσιάσουμε τη σύγκλιση της primal και dual residual της γεννήτριας 1:



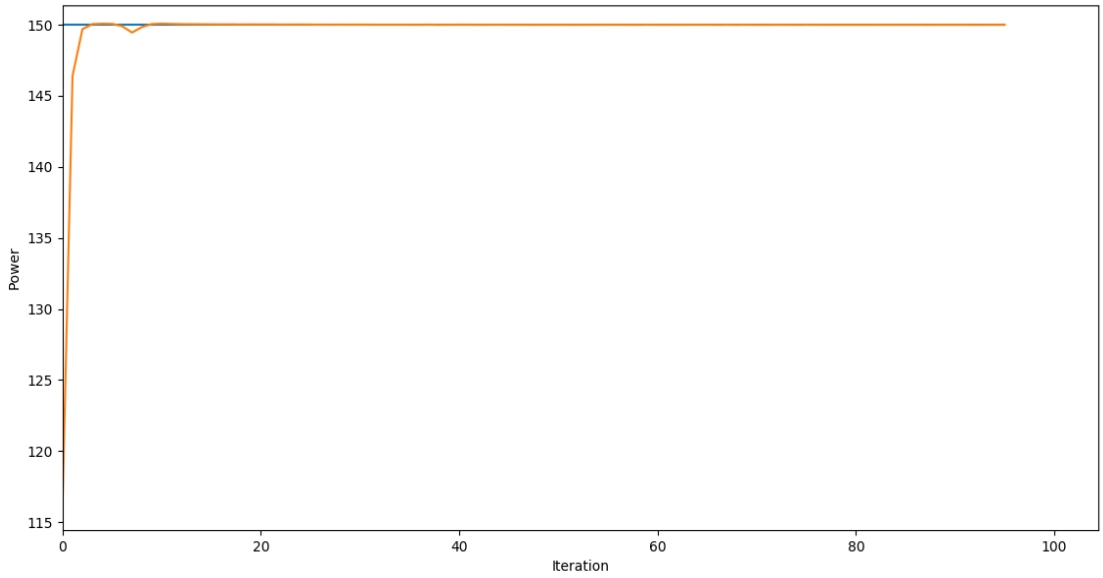

Διάγραμμα 8.7: Η εξέλιξη της primal residual της πρώτης γεννήτριας κατά τη διάρκεια αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών)




Διάγραμμα 8.8: Η εξέλιξη της dual residual της πρώτης γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών)

Τέλος στο διάγραμμα 8.9 που ακολουθεί παρατηρούμε την ισορροπία μεταξύ της παραγωγής και της ζήτησης κατά τη διάρκεια του αλγορίθμου. Συγκεκριμένα, όπως και

πριν βλέπουμε πως η ισορροπία παραγωγής-ζήτησης επιτυγχάνεται μετά τις πρώτες 16 επαναλήψεις του αλγορίθμου.




Διάγραμμα 8.9: Η εξέλιξη της ισορροπίας μεταξύ παραγωγής και ζήτησης κατά τη διάρκεια εκτέλεσης του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών)

### 8.3.1 Σύγκριση αποτελεσμάτων κεντρικού και αποκεντρωμένου αλγόριθμο

Ο τύπος του σχετικού σφάλματος είναι ο ακόλουθος :

$$\Sigma = \frac{|x_\pi - x_A|}{x_A} \cdot 100\% \quad (8.1)$$

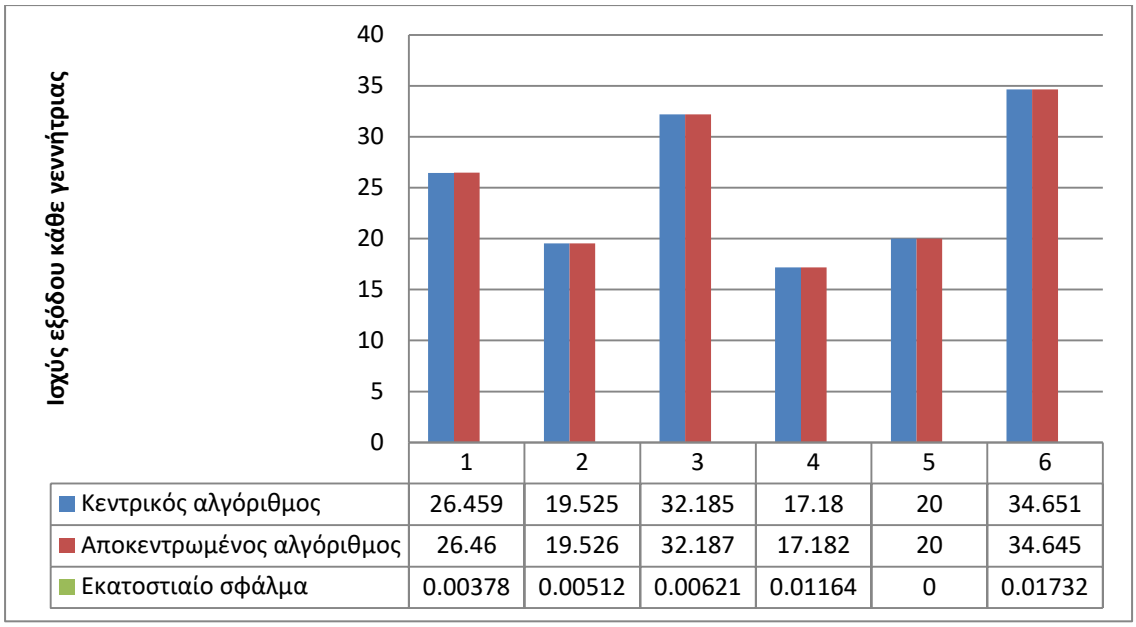
Όπου :

$x_A$: η πραγματική τιμή του μεγέθους που μετρούμε (εν προκειμένη περιπτώσει το $x_A$ είναι η ισχύς εξόδου κάθε γεννήτριας όπως προκύπτει από τον κεντρικό αλγόριθμο)

Και

$x_\pi$: η πειραματική τιμή (εν προκειμένη περιπτώσει το $x_\pi$ είναι η ισχύς εξόδου κάθε γεννήτριας όπως προκύπτει από τον αποκεντρωμένο αλγόριθμο)

Στο διάγραμμα 8.10 που ακολουθεί μπορούμε να δούμε παραστατικά τη σύγκριση μεταξύ κεντρικού και αποκεντρωμένου αλγορίθμου (γενικά ο κεντρικός αλγόριθμος χρησιμοποιείται σε όλη την έκταση της διπλωματικής σαν πρότυπος αλγόριθμος με βάση τον οποίο συγκρίνουμε την ορθότητα των αποτελεσμάτων μας).




| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Κεντρικός αλγόριθμος | 26.459 | 19.525 | 32.185 | 17.18 | 20 | 34.651 |
| Αποκεντρωμένος αλγόριθμος | 26.46 | 19.526 | 32.187 | 17.182 | 20 | 34.645 |
| Εκατοστιαίο σφάλμα | 0.00378 | 0.00512 | 0.00621 | 0.01164 | 0 | 0.01732 |

Διάγραμμα 8.10: Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ε=0)

Από το διάγραμμα 8.10 παρατηρούμε πως ο αποκεντρωμένος αλγόριθμος δεν διαφέρει καθόλου από τον κεντρικό. Στον κεντρικό αλγόριθμο όλες οι πράξεις εκτελούνται από τον κώδικα της python που έχει μεγάλη ακρίβεια σε δεκαδικά ψηφία. Αντιθέτως όταν χρησιμοποιείται ο αποκεντρωμένος αλγόριθμος κάθε κόμβος εκτελεί τοπικά τους υπολογισμούς του με πολλά δεκαδικά ψηφία αλλά το αποτέλεσμα που παράγει το στέλνει στο έξυπνο συμβόλαιο πολλαπλασιασμένο επί 1000, δηλαδή κρατιέται ακρίβεια τριών δεκαδικών. Επίσης όλοι οι υπόλοιποι υπολογισμοί στο έξυπνο συμβόλαιο γίνονται κρατώντας ακρίβεια τριών δεκαδικών ψηφίων με αποτέλεσμα να χάνουμε σε ακρίβεια. Παρόλα αυτά το σφάλματα μεταξύ των τιμών που προκύπτουν από τους δύο αλγορίθμους είναι μηδαμινό (υπάρχει ένα μέσοι σφάλμα 0,00735%). Έτσι μπορούμε με ασφάλεια να ισχυριστούμε ότι ο αποκεντρωμένος αλγόριθμος είναι έγκυρος.

### 8.3.2 Κατανάλωση gas, υπόλοιπο χρηστών και έξοδα του κάθε χρήστη

Όπως έχει αναφερθεί και στην ενότητα 3.6.6 οι συναλλαγές στο Ethereum έχουν κάποιο κόστος (gas). Δηλαδή όταν κάποιος στέλνει δεδομένα που αποθηκεύονται στο blockchain (άρα καταλαμβάνουν αποθηκευτικό χώρο) ή κάνει συναλλαγές που αλλάζουν την κατάσταση (state) της Εικονικής Μηχανής του Ethereum (ουσιαστικά όταν "γράφει" δεδομένα στο έξυπνο συμβόλαιο, όχι όταν απλά διαβάζει) και άρα γίνονται υπολογισμοί από την Εικονική Μηχανή, τότε πληρώνει κάποια ether για κάθε υπολογιστικό βήμα που εκτελείται . Έτσι λοιπόν μας ενδιαφέρει να ξέρουμε πόσο gas και επομένως πόσα ether, καταναλώνει κάθε κόμβος που συμμετέχει στο πρόβλημα Οι-

κονομικής Κατανομής Φορτίου. Σκόπιμο κρίνεται να αναφερθεί ότι στο Ganache η προεπιλεγμένη (default) τιμή για το gas price είναι 20gWei/gas.

Στον πίνακα που ακολουθεί μπορούμε να δούμε το υπόλοιπο (σε ether) κάθε κόμβου, το ποσό (σε ether) που ξόδεψε κατά τη διάρκεια εκτέλεσης του αλγορίθμου καθώς και την κατανάλωση του σε gas:

| Κόμβος | Υπόλοιπο(ether) κάθε κόμβου μετά το πέρας του αλγορίθμου | Ποσό (ether) που ξόδεψε κάθε κόμβος κατά τη διάρκεια του αλγορίθμου | Κατανάλωση gas κάθε κόμβου | Ελάχιστη κατανάλωση gas | Μέγιστη κατανάλωση gas | Μέση κατανάλωση gas |
|---|---|---|---|---|---|---|
| 1 | 99,8915057 | 0,10849426 | 5.424.713 | 3.719.291 | 58.127.235 | 13.182.518,67 |
| 2 | 99,9256142 | 0,07438582 | 3.719.291 | | | |
| 3 | 99,9233935 | 0,07660646 | 3.830.323 | | | |
| 4 | 99,9211742 | 0,07882582 | 3.941.291 | | | |
| 5 | 99,9189548 | 0,08104518 | 4.052.259 | | | |
| 6 | 98,8374553 | 1,1625447 | 58.127.235 | | | |

Πίνακας 8.4 :Υπόλοιπο λογαριασμού κάθε κόμβου, κατανάλωση gas και ether (για σύστημα 6 γεννητριών και gas price=20gWei/gas)

Από τον πίνακα 8.4 παρατηρούμε ότι η μέση κατανάλωση gas είναι 13.182.518,67 . Οι κόμβοι 2-5 έχουν περίπου την ίδια κατανάλωση gas ενώ παραπάνω έχει ο κόμβος 1 και πολύ παραπάνω ο κόμβος 6. Ο κόμβος 1 έχει παραπάνω κατανάλωση gas διότι είναι ο κόμβος που δημιουργεί και αναπτύσσει (deploy) στο Ganache το έξυπνο συμβόλαιο. Κάνει μια συναλλαγή παραπάνω σε σχέση με τους υπόλοιπους κόμβους γι' αυτό και καταναλώνει λίγα περισσότερα gas. Όσον αφορά τον κόμβο 6 παρατηρούμε ότι η κατανάλωση του σε gas είναι μεγαλύτερη σε σχέση με τους άλλους κόμβους. Αυτό εξηγείται αν αναλογιστούμε τον τρόπο με τον οποίο δουλεύει το έξυπνο συμβόλαιο. Το έξυπνο συμβόλαιο δέχεται από όλους τους κόμβους τις λύσεις τους $x_i(k+1)$ και αφού δεχτεί τις λύσεις από όλους τους κόμβους, τότε αυτόματα εκτελείται ο κώδικας που υπολογίζει αρχικά το $\eta$ και στη συνέχεια τα $y_i(k+1)$ και $\lambda_i(k+1)$ αλλά ελέγχει επίσης και αν έχει επιτευχτεί σύγκλιση, βλέποντας αν η primal και dual residual κάθε κόμβου τείνει στο μηδέν. Δηλαδή το συμβόλαιο εκτελεί το βασικό του αλγόριθμο (που είναι ο υπολογισμός του aggregation step και ο έλεγχος σύγκλισης) μόνο εφόσον έχει "αντιληφθεί" ότι έχουν στείλει όλοι οι κόμβοι τα $x_i(k+1)$ που έχουν υπολογίσει και όταν στείλει ο τελευταίος, τότε ενεργοποιείται ο εν λόγω κώδικας. Οπότε ο κόμβος που θα στείλει τελευταίος τα δεδομένα του στο blockchain θα είναι αυτός που θα "χρεωθεί" όλα τα υπολογιστικά βήματα που απαιτούνται για τον υπολογισμό του $\eta$, των $y_i(k+1)$ και $\lambda_i(k+1)$ αλλά και τον έλεγχο για το αν έχει επιτευχθεί σύγκλιση. Στη παρούσα διπλωματική έχει οριστεί αυθαίρετα ότι οι κόμβοι θα στέλνουν τις λύσεις τους στο blockchain με πρώτο τον 1 και τελευταίο τον 6 (με τη σειρά που οι κόμβοι καταχωρούνται στην whitelist). Γι αυτό και ο κόμβος 6 (και

γενικά ο εκάστοτε τελευταίος κόμβος της whitelist) σε όλα τα σενάρια θα είναι ο κόμβος που θα "καταναλώνει" το περισσότερο gas, άρα και ether και συνεπώς και €.

Ας δούμε όμως και τα χρήματα που ξόδεψε κάθε κόμβος σε ευρώ(€) ώστε να έχουμε μία καλύτερη εικόνα . Με την τωρινή αντιστοίχιση ισχύει:

$$1\ ether \rightarrow 168{,}57\ €$$

Έτσι λοιπόν ο κάθε κόμβος χάλασε σε € :

| Κόμβος | Ποσό που ξόδεψε κάθε κόμβος κατά τη διάρκεια του αλγορίθμου σε (€) | Ελάχιστο ποσό (€) | Μέγιστο ποσό (€) | Μέση τιμή (€) |
|---|---|---|---|---|
| 1 | 18,29 | 12,54 | 195,97 | 44,44 |
| 2 | 12,54 | | | |
| 3 | 12,91 | | | |
| 4 | 13,29 | | | |
| 5 | 13,66 | | | |
| 6 | 195,97 | | | |

Πίνακας 8.5: Ποσό που ξόδεψε κάθε κόμβος σε € (για σύστημα 6 γεννητριών και gas price=20gWei/gas)

Όπως παρατηρούμε από τον παραπάνω πίνακα κατά μέσο όρο κάθε κόμβος που συμμετέχει στο πρόβλημα Οικονομικής Κατανομής Φορτίου μετά το πέρας του αλγορίθμου έχει χαλάσει περίπου 44,44€ (βασικά οι κόμβοι 2-5 έχουν χαλάσει περίπου 13€ έκαστος, απλά ο πρώτος κόμβος και ειδικά ο έκτος ανεβάζουν πολύ το μέσο όρο) . Η αλήθεια είναι ότι το κόστος αυτό είναι απαγορευτικό για να μπορέσει να εφαρμοστεί στην πραγματικότητα. Ας έχουμε όμως υπόψη μας ότι το gas price που είναι 20 gWei/gas είναι αρκετά υψηλό. Η τυπική τιμή του gas price είναι περίπου 3-4 gWei/gas για τέτοιες εφαρμογές. Θα επαναλάβουμε λοιπόν ξανά το παραπάνω σενάριο με gas price 3gWei/gas ώστε να έχουμε μια εικόνα που είναι πιο κοντά στην πραγματικότητα. Όπως είναι προφανές το αποτέλεσμα του προβλήματος οικονομικής κατανομής φορτίου (κατανομή ισχύος στις γεννήτριες) δεν αλλάζει αλλά θα αλλάξει προφανώς το ποσό (σε ether) που ξόδεψε κάθε κόμβος κατά τη διάρκεια εκτέλεσης του αλγορίθμου, το υπόλοιπό του καθώς και τα αντίστοιχα € που σπατάλησε κάθε κόμβος. Πρέπει να τονίσουμε ότι η κατανάλωση gas κάθε κόμβου δεν αλλάζει με την αλλαγή του gas price από 20 σε 3 gWei/gas. Αυτό συμβαίνει διότι η κατανάλωση gas έχει να κάνει με τα υπολογιστικά βήματα που εκτελεί το virtual machine του ethereum ώστε να επεξεργαστεί τα δεδομένα που του στείλαμε και από τη στιγμή που ο κάθε κόμβος στο συγκεκριμένο σενάριο στέλνει τα ίδια δεδομένα (μιας και δεν έχει αλλάξει κάτι στο σύστημα παρά μόνο το gas price), η κατανάλωση σε gas θα είναι η ίδια αλλά θα είναι πολύ μικρότερα τα έξοδα του σε ether, άρα και σε €. Στον πίνακα 8.6 που ακολουθεί βλέπουμε τα υπόλοιπο κάθε κόμβου, το ποσό που ξόδεψε σε ether

(αλλά και σε € μέσω της ισοτιμίας που έχει γραφτεί παραπάνω) με gas price =3 gWei/gas:

| Κόμβος | Υπόλοιπο(Ether) κάθε κόμβο μετά το πέρας του αλγορίθμου | Ποσό (Ether)που ξόδεψε κάθε κόμβος κατά τη διάρκεια του αλγορίθμου | Ποσό (€)που ξόδεψε κάθε κόμβος κατά τη διάρκεια του αλγορίθμου | Ελάχιστο ποσό (€) | Μέγιστο ποσό (€) | Μέση τιμή (€) |
|---|---|---|---|---|---|---|
| 1 | 99,9837259 | 0,01627414 | 2,74 | 1,88 | 29,4 | 6,67 |
| 2 | 99,9888421 | 0,01115787 | 1,88 | | | |
| 3 | 99,9885090 | 0,01149097 | 1,94 | | | |
| 4 | 99,9881761 | 0,01182387 | 1,99 | | | |
| 5 | 99,9878432 | 0,01215678 | 2,05 | | | |
| 6 | 99,8256183 | 0,1743817 | 29.4 | | | |

Πίνακας 8.6: Υπόλοιπο λογαριασμού κάθε κόμβου και ποσό που σπαταλήθηκε (σε ether και €) (για σύστημα 6 γεννητριών και gas price=3gWei/gas)

Από τον πίνακα 8.6 παρατηρούμε ότι με gas price= 3 gWei/gas η μέση κατανάλωση χρημάτων περίπου 6.67 €. Σε σχέση με την περίπτωση που το gas price έχει την προεπιλεγμένη τιμή 20 gWei/gas έχουμε μικρότερη μέση κατανάλωση χρημάτων κατά 85%. Βέβαια και πάλι η μέση κατανάλωση χρημάτων είναι αρκετά υψηλή . Στην επόμενη ενότητα θα προσπαθήσουμε να μειώσουμε τον αριθμό των επαναλήψεων που απαιτούνται για σύγκλιση έτσι ώστε να μειωθεί η κατανάλωση gas και χρημάτων ανά κόμβο.

### 8.3.3 Εκτέλεση του αποκεντρωμένου αλγορίθμου για διαφορετικά ε

Όπως ήδη έχει αναφερθεί στην ενότητα 5.2 , ο αλγόριθμος περατώνεται όταν ισχύει :

$$|r_i(k)| \leq \varepsilon \Rightarrow |x_i(k) - y_i(k)| \leq \varepsilon \, για \, κάθε \, i = 1,2, \dots, N$$

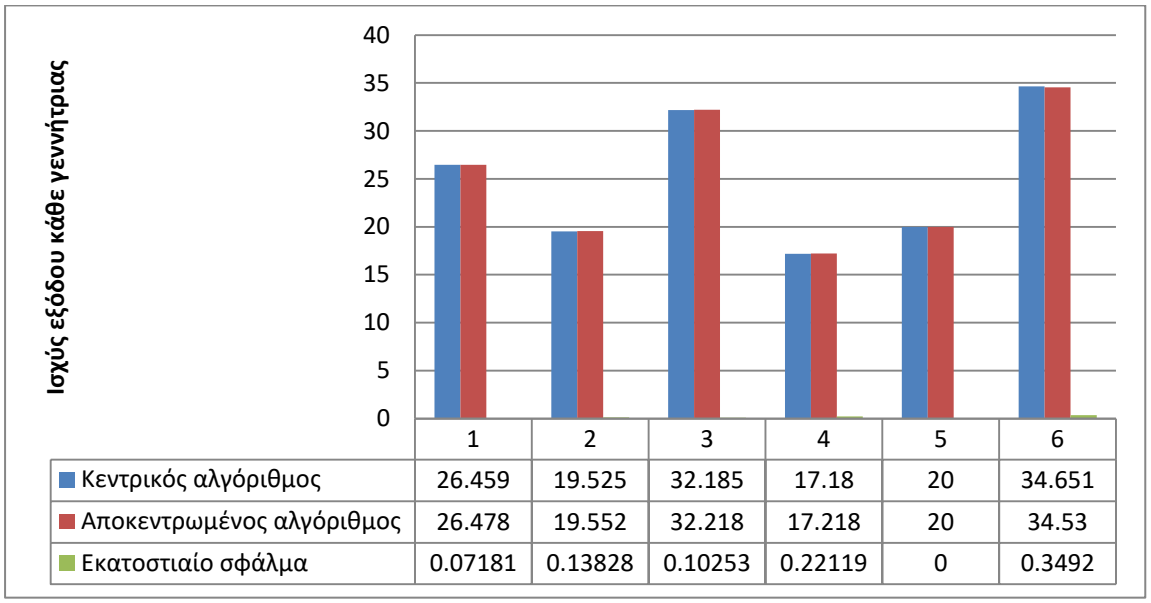
$$|s_i(k)| \leq \varepsilon \Rightarrow |-\rho \cdot (y_i(k) - y_i(k-1))| \leq \varepsilon \, για \, κάθε \, i = 1,2, \dots, N$$

Όπου $\varepsilon$ μικρός αριθμός

Έτσι λοιπόν μεταβάλλοντας το $\varepsilon$ θα εξετάσουμε πως μεταβάλλεται ο αριθμός των επαναλήψεων που απαιτούνται για σύγκλιση, πως μεταβάλλεται η μέση κατανάλωση gas αλλά και το μέσο οικονομικό κόστος κάθε κόμβου. Θα πρέπει όμως να έχουμε υπόψη ότι το έξυπνο συμβόλαιο υπολογίζει τα $x_i$, $y_i$,$\lambda_i$ πολλαπλασιασμένα επί 1000 . Όποτε τα ε που ορίζουμε θα πρέπει να είναι πολλαπλασιασμένα επί 1000. Μελετάμε λοιπόν τις ακόλουθες περιπτώσεις :

**ε=10 (0,01)**

Εκτελούμε τον αλγόριθμο και έχουμε σύγκλιση μετά από 58 επαναλήψεις. Δηλαδή ο αλγόριθμος συγκλίνει γρηγορότερα σε σχέση με πριν που είχαν απαιτηθεί 96 επαναλήψεις. Στο διάγραμμα 8.11 που ακολουθεί μπορούμε να δούμε τις τιμές ισχύος όπως προκύπτουν από τον κεντρικό αλγόριθμο αλλά και από τον αποκεντρωμένο με ε=10 καθώς τα σχετικά σφάλματα.




| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Κεντρικός αλγόριθμος | 26.459 | 19.525 | 32.185 | 17.18 | 20 | 34.651 |
| Αποκεντρωμένος αλγόριθμος | 26.478 | 19.552 | 32.218 | 17.218 | 20 | 34.53 |
| Εκατοστιαίο σφάλμα | 0.07181 | 0.13828 | 0.10253 | 0.22119 | 0 | 0.3492 |

Διάγραμμα 8.11 : Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ε=10)

Όπως παρατηρούμε παρότι μεγαλώσαμε το όριο σύγκλισης δεν χάνουμε ιδιαίτερα σε ακρίβεια και έχουμε αρκετά ικανοποιητική ακρίβεια (ο αποκεντρωμένος αλγόριθμος με ε=10 υπολογίζει τις τιμές ισχύος των γεννητριών με ένα μέσο σφάλμα της τάξης 0,147%).

Στη συνέχεια θα εξετάσουμε την κατανάλωση gas κάθε χρήστη ώστε να δούμε πως άλλαξε σε σχέση με την αύξηση του ε και άρα τη μείωση των επαναλήψεων σύγκλισης. Στο πίνακα 6.9 που ακολουθεί βλέπουμε  κατανάλωση gas κάθε χρήστη, και το ποσό που ξόδεψε σε ether και σε € :

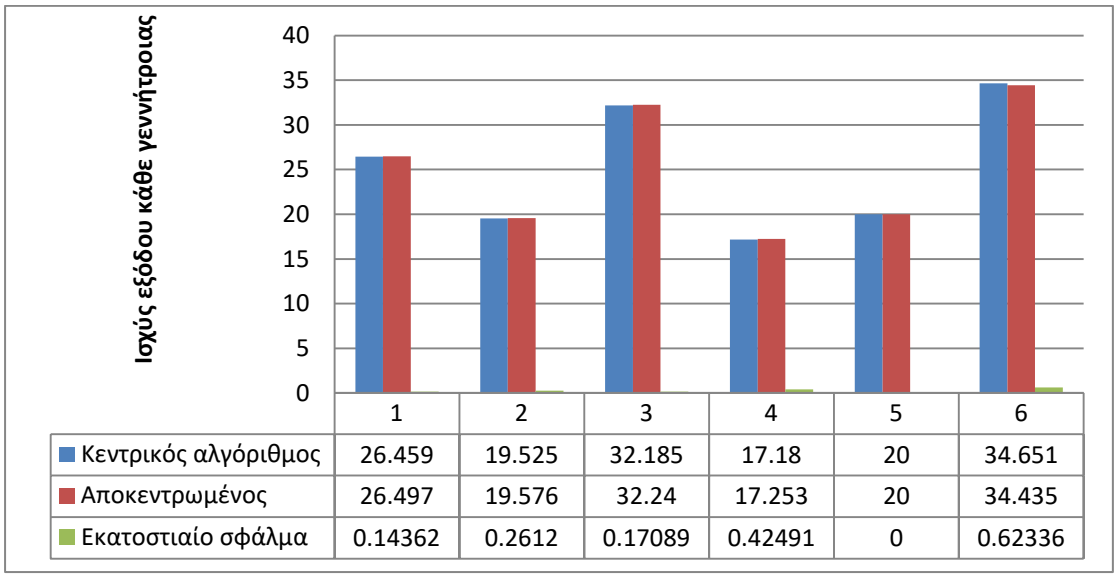
| Κατανάλωση gas ανά κόμβο | Ποσό που ξόδεψε κάθε κόμβος σε ether | Ποσό που ξόδεψε κάθε κόμβος σε € | Μέση Χρηματική Κατανάλωση σε € | Μέση κατανάλωση gas |
|---|---|---|---|---|
| 4.036.297 | 0,01210889 | 2,041 | 3,92 | 8.145.332,17 |
| 2.269.971 | 0,00680991 | 1,148 | | |
| 2.337.531 | 0,00701259 | 1,182 | | |
| 2.405.027 | 0,00721508 | 1,182 | | |
| 2.472.522 | 0,00741757 | 1,25 | | |
| 35.350.645 | 0,10605194 | 17,877 | | |

Πίνακας 8.7: Κατανάλωση gas κάθε κόμβου και ποσό που σπαταλήθηκε σε ether και € (για σύστημα 6 γεννητριών και ε=10 )

Όπως παρατηρούμε η μέση χρηματική κατανάλωση έπεσε κατά 70.15% σε σχέση με την περίπτωση που το ε είναι 0 και αυτό χωρίς να χάνουμε ιδιαίτερα σε ακρίβεια.

## ε=20 (0.02)

Ακολουθώντας την ίδια λογική με πριν θέτουμε το ε =20 και εκτελούμε τον αποκεντρωμένο αλγόριθμο. Το πρόβλημα οικονομικής κατανομής φορτίου λύνεται μετά από 51 επαναλήψεις. Στο διάγραμμα 8.12 που ακολουθεί μπορούμε να δούμε τις τιμές ισχύος όπως προκύπτουν από τον κεντρικό αλγόριθμο αλλά και από τον αποκεντρωμένο με ε=20 καθώς τα σχετικά σφάλματα.:




| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Κεντρικός αλγόριθμος | 26.459 | 19.525 | 32.185 | 17.18 | 20 | 34.651 |
| Αποκεντρωμένος | 26.497 | 19.576 | 32.24 | 17.253 | 20 | 34.435 |
| Εκατοστιαίο σφάλμα | 0.14362 | 0.2612 | 0.17089 | 0.42491 | 0 | 0.62336 |

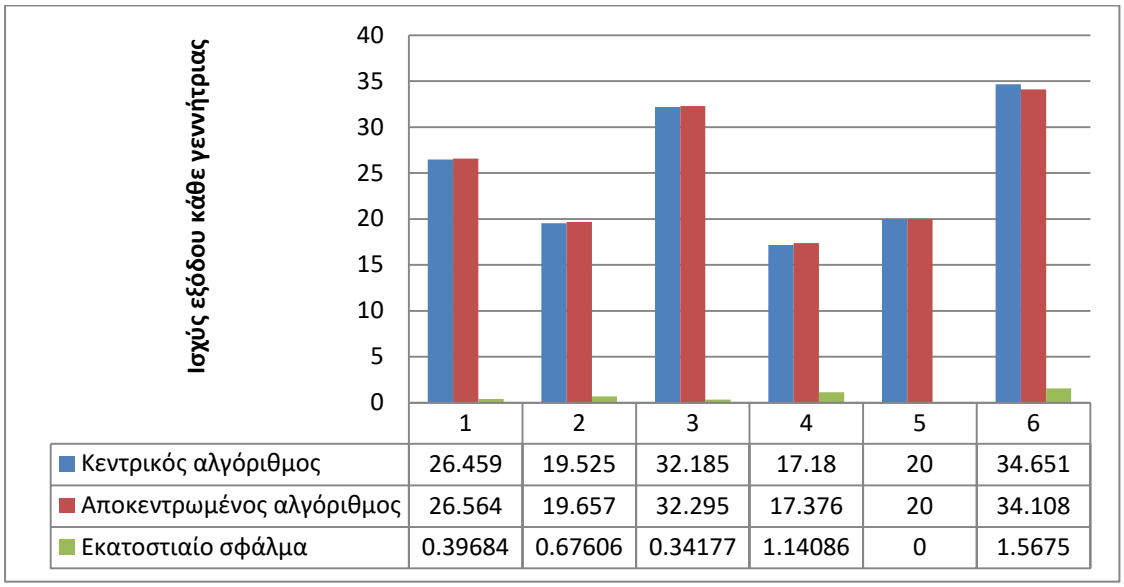
Διάγραμμα 8.12: Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ε=20 )

Παρατηρούμε λοιπόν ότι και για ε=20 τα αποτελέσματα δεν διαφέρουν πολύ από τα αντίστοιχα που προκύπτουν από τον κεντρικό αλγόριθμο (ο αποκεντρωμένος αλγόριθμος με ε=20 υπολογίζει τις τιμές ισχύος των γεννητριών με ένα μέσο σφάλμα της

τάξης 0,27%). Επίσης στη συγκεκριμένη περίπτωση η μέση κατανάλωση gas είναι 7.216.156,33 και η μέση χρηματική κατανάλωση είναι 3,64€ (0,0216 ether).

**ε=50 (0.05)**

Εκτελούμε τον αλγόριθμο με ε=50 και καταλήγουμε σε σύγκλιση μετά από 40 επαναλήψεις. Στο διάγραμμα 8.13 που ακολουθεί μπορούμε να δούμε τις τιμές ισχύος όπως προκύπτουν από τον κεντρικό αλγόριθμο αλλά και από τον αποκεντρωμένο με ε=50 καθώς τα σχετικά σφάλματα.



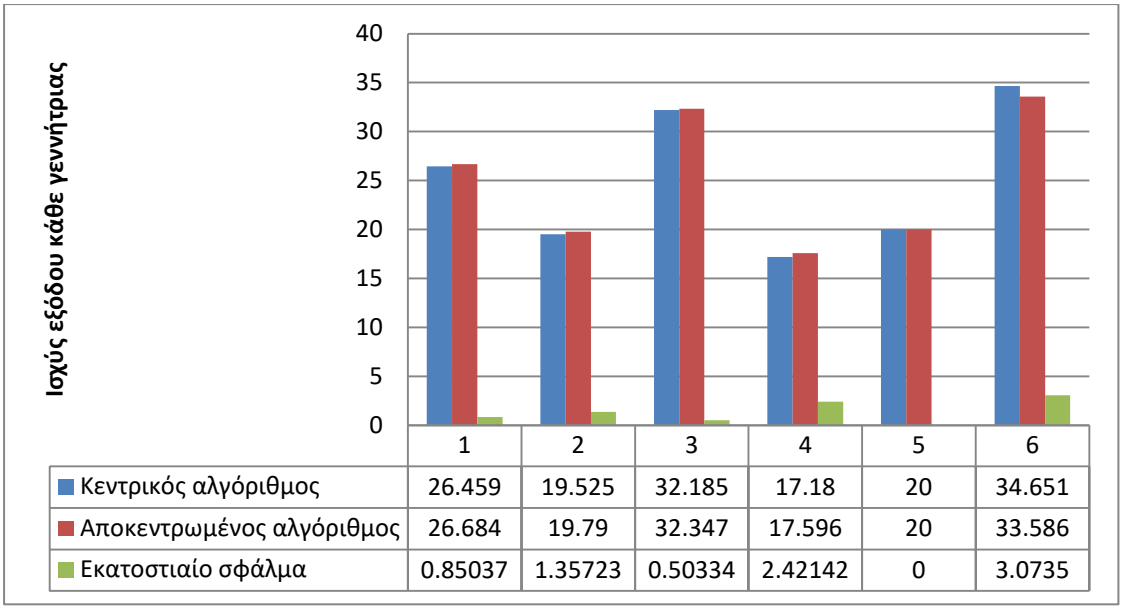

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Κεντρικός αλγόριθμος | 26.459 | 19.525 | 32.185 | 17.18 | 20 | 34.651 |
| Αποκεντρωμένος αλγόριθμος | 26.564 | 19.657 | 32.295 | 17.376 | 20 | 34.108 |
| Εκατοστιαίο σφάλμα | 0.39684 | 0.67606 | 0.34177 | 1.14086 | 0 | 1.5675 |

Διάγραμμα 8.13: Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ε=50)

Παρατηρούμε ότι οι τιμές ισχύος της $4^{ης}$ και $6^{ης}$ γεννήτριας διαφέρουν αρκετά από τις αντίστοιχες τιμές του που είναι υπολογισμένες με τον κεντρικό αλγόριθμο (ο αποκεντρωμένος αλγόριθμος με ε=50 υπολογίζει τις τιμές ισχύος των γεννητριών με ένα μέσο σφάλμα της τάξης 0,69%). Η μέση κατανάλωση gas είναι 5.748.493 και η μέση χρηματική κατανάλωση είναι 2,90€ (0,0172 ether).

**ε=100 (0.1)**

Τέλος δοκιμάζουμε και για ε=100 το οποίο είναι αρκετά μεγάλο ε. Εκτελούμε λοιπόν τον αποκεντρωμένο αλγόριθμο και καταλήγουμε σε σύγκλιση μετά από 32 επαναλήψεις. Στο διάγραμμα 6.14 που ακολουθεί μπορούμε να δούμε και τη σύγκριση του αποκεντρωμένου αλγορίθμου για ε=100 με τον κεντρικό αλγόριθμο:



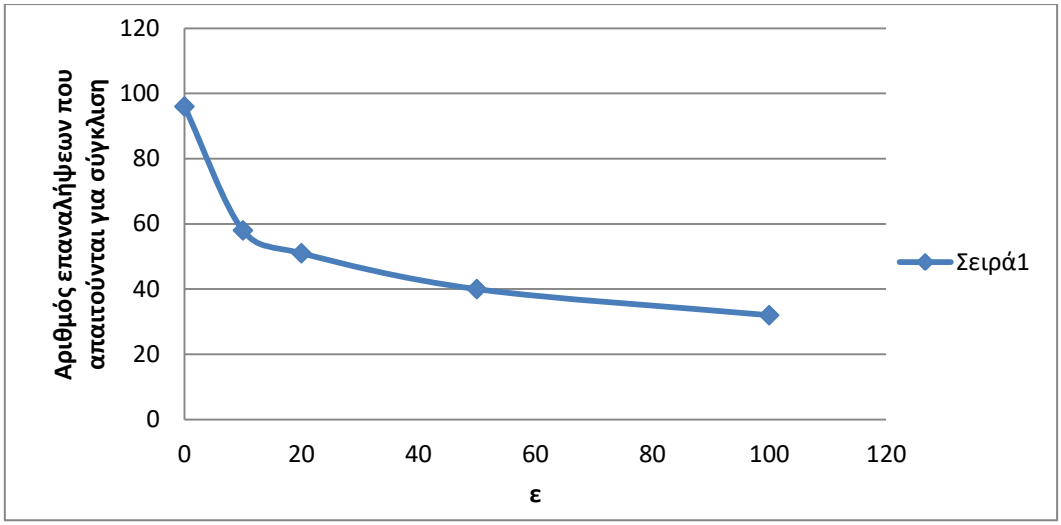

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Κεντρικός αλγόριθμος | 26.459 | 19.525 | 32.185 | 17.18 | 20 | 34.651 |
| Αποκεντρωμένος αλγόριθμος | 26.684 | 19.79 | 32.347 | 17.596 | 20 | 33.586 |
| Εκατοστιαίο σφάλμα | 0.85037 | 1.35723 | 0.50334 | 2.42142 | 0 | 3.0735 |

Διάγραμμα 8.14: Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ε=100)

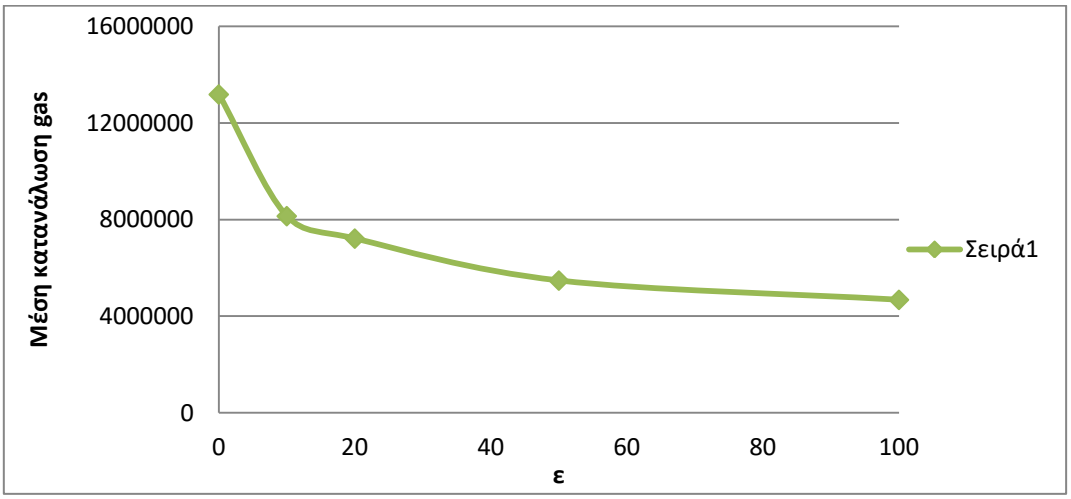
Για ε=100 παρατηρούμε ότι έχουμε μεγάλο σφάλμα κυρίως στην παραγωγή της γεννήτριας 6 αλλά και της 4 (ο αποκεντρωμένος αλγόριθμος με ε=100 υπολογίζει τις τιμές ισχύος των γεννητριών με ένα μέσο σφάλμα της τάξης 1,37%). Τέλος αναφέρουμε ότι η μέση κατανάλωση gas πλέον είναι 4.680308,33 ενώ η μέση χρηματική κατανάλωση είναι 2,36€ (0,014 ether).

### 8.3.4 Αποτελέσματα του αποκεντρωμένου αλγορίθμου με διαφορετικά ε

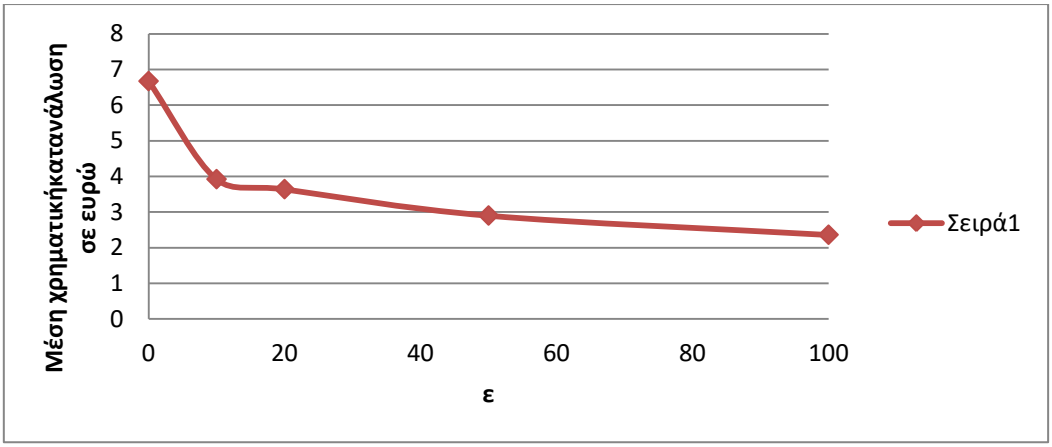
Στο διαγράμματα 8.15 ,8.16, 8.17 και 8.18 βλέπουμε πως μεταβάλλοντας το ε από 0 έως 100, μεταβάλλεται ο αριθμός των επαναλήψεων που απαιτούνται για σύγκλιση , η μέση κατανάλωση gas  και χρημάτων αλλά και το μέσο σφάλμα με τις οποίες κάθε φορά ο αποκεντρωμένος αλγόριθμος υπολογίζει τις τιμές ισχύος των γεννητριών.



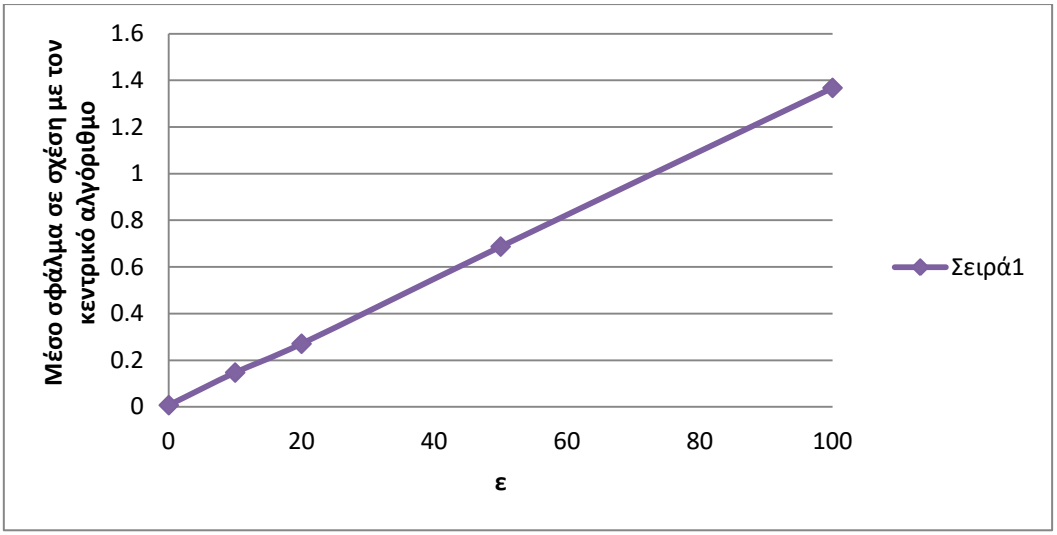

Διάγραμμα 8.15: Αριθμός των επαναλήψεων που απαιτούνται για σύγκλιση σε συνάρτηση με το ε




Διάγραμμα 8.16: Μέση κατανάλωση gas σε συνάρτηση με το ε




Διάγραμμα 8.17: Μέση χρηματική κατανάλωση (€)σε συνάρτηση με το ε




Διάγραμμα 8.18: Μέσο σφάλμα σε σχέση με τον κεντρικό αλγόριθμο συναρτήσει του ε

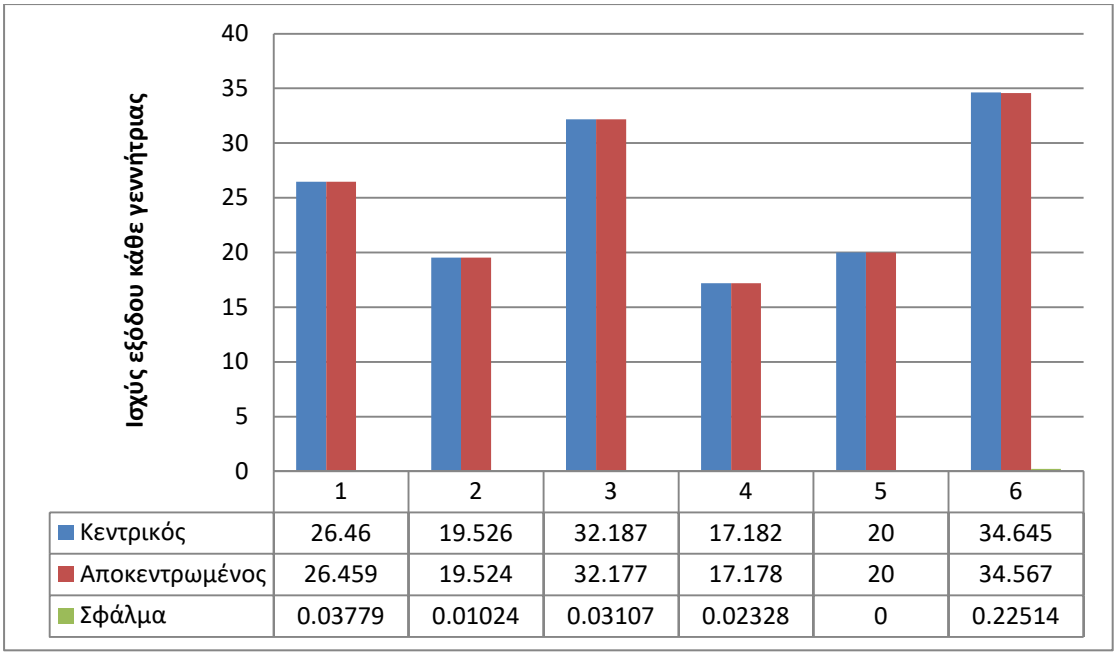
Παρατηρούμε ότι το μέσο σφάλμα σε σχέση με το κεντρικό αλγόριθμο είναι μια αρκετά γραμμική συνάρτηση. Δηλαδή όσο αυξάνει το ε τόσο θα αυξάνεται και η απόκλιση των τιμών ισχύος που υπολογίζει ο αποκεντρωμένος αλγόριθμος σε σχέση με τις αντίστοιχες τιμές που υπολογίζονται από τον κεντρικό. Όσο αυξάνεται το όριο σύγκλισης τόσο μειώνεται η κατανάλωση gas κάθε κόμβου και αντίστοιχα και τα χρήματα που σπαταλάει αλλά βέβαια οι τιμές που προκύπτουν αρχίζουν να διαφοροποιούνται όλο και περισσότερο σε σχέση με τον κεντρικό αλγόριθμο (που αποδίδει το βέλτιστο αποτέλεσμα) .

### 8.3.5 Εκτέλεση του αποκεντρωμένου αλγορίθμου για διαφορετικά ρ

Σε αυτή την ενότητα θα μεταβάλλουμε την παράμετρο $\rho$ και θα εξετάσουμε πως συμπεριφέρεται ο αποκεντρωμένος αλγόριθμος. Συγκεκριμένα θέλουμε να δούμε πως μεταβάλλεται ο αριθμός των επαναλήψεων που απαιτούνται για σύγκλιση, η μέση χρηματική κατανάλωση αλλά και η μέση κατανάλωση gas αλλά θέλουμε κάθε φορά να συγκρίνουμε τα αποτελέσματα μας με τα αποτελέσματα του κεντρικού αλγορίθμου (βέλτιστα αποτελέσματα λόγω της μεγάλης ακρίβειας της python και του $\varepsilon$ που είχε επιλεγεί $10^{-14}$). Σε όλες τις περιπτώσεις που ακολουθούν το $\varepsilon$ σύγκλισης έχει επιλεγεί 10 (0,01) .

$$\underline{\boldsymbol{\rho = 0,1}}$$

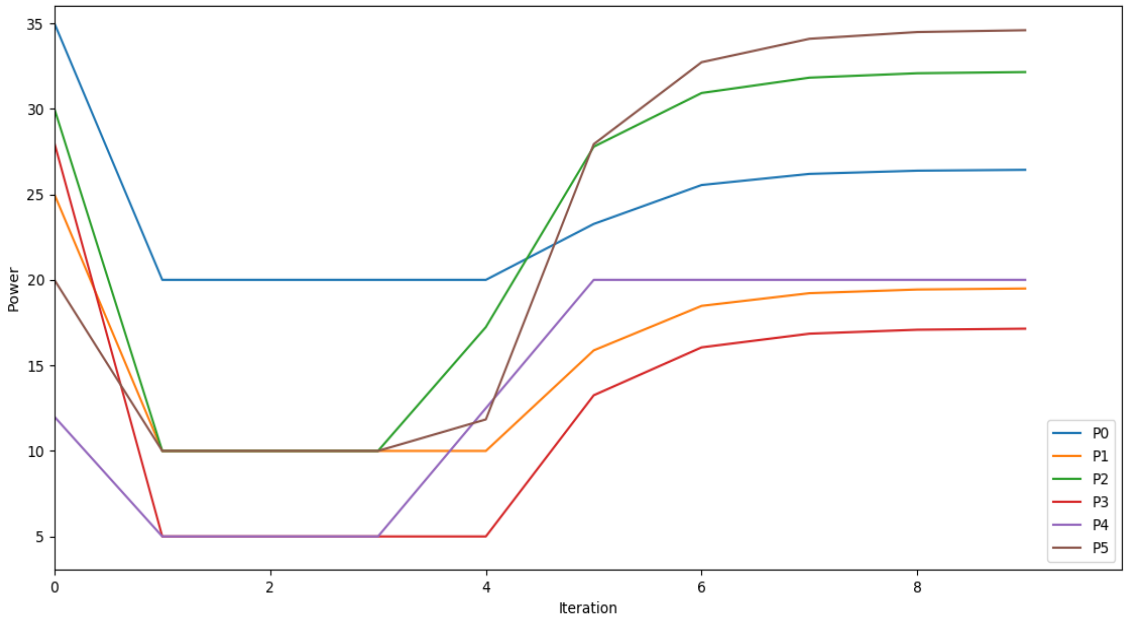
Ορίζοντας το ρ =0,1, (όπως έχει εξηγηθεί στο κεφάλαιο 7 ο δημιουργός του συμβολαίου περνάει την τιμή στο συμβόλαιο ως 1/0,1 δηλαδή ρ=10) εκτελούμε τον αποκεντρωμένο αλγόριθμο και μετά από 14 επαναλήψεις συγκλίνει. Μέχρι στιγμής αυτή είναι η γρηγορότερη σύγκλιση του αλγορίθμου. Στο διάγραμμα 8.19 που ακολουθεί μπορούμε να δούμε πέρα από τις τιμές ισχύος που προκύπτουν για κάθε γεννήτρια και μία σύγκριση με τον κεντρικό αλγόριθμο.




| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Κεντρικός | 26.46 | 19.526 | 32.187 | 17.182 | 20 | 34.645 |
| Αποκεντρωμένος | 26.459 | 19.524 | 32.177 | 17.178 | 20 | 34.567 |
| Σφάλμα | 0.03779 | 0.01024 | 0.03107 | 0.02328 | 0 | 0.22514 |

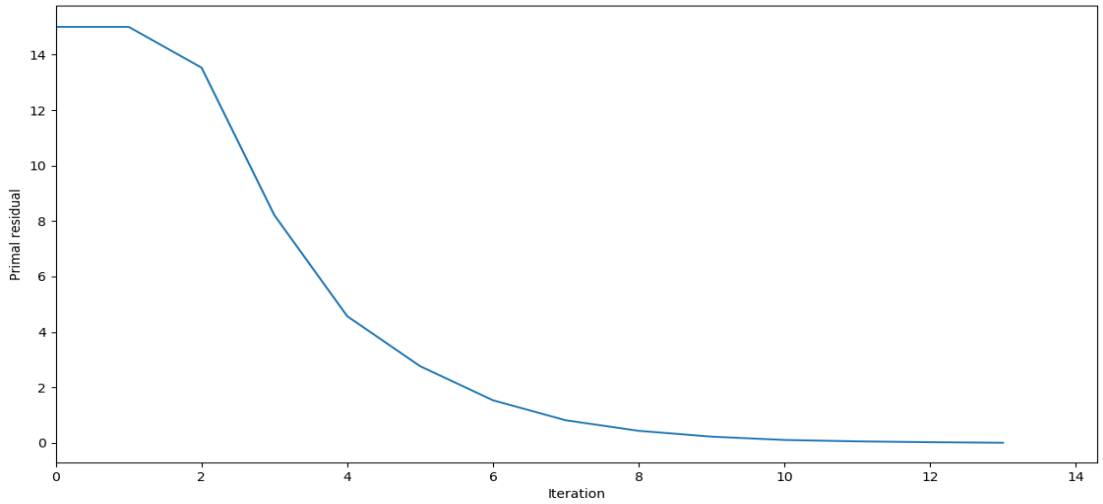
Διάγραμμα 8.19: Σύγκριση αποκεντρωμένου αλγορίθμου με $\rho = 0{,}1$ με τον κεντρικό αλγόριθμο (για σύστημα 6 γεννητριών)

Παρατηρούμε ότι οι τιμές ισχύος που προκύπτουν από τον αποκεντρωμένο αλγόριθμο με $\rho = 0{,}1$ δεν διαφέρουν πολύ από αυτές που προκύπτουν από τον κεντρικό αλγόριθμο (υπάρχει ένα μέσο σφάλμα περίπου 0,0489%). Στο διάγραμμα 8.20 που ακολουθεί μπορούμε να δούμε πως η ισχύς κάθε γεννήτριας εξελίσσεται με την πάροδο του αλγορίθμου μέχρι να σταθεροποιηθεί στη τελική τιμή της .

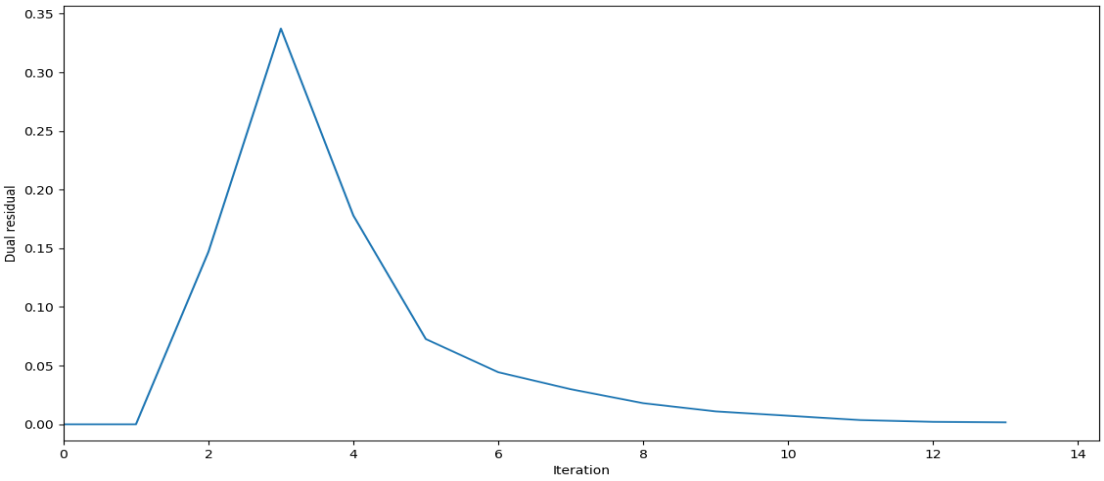



Διάγραμμα 8.20: Η εξέλιξη της ισχύος εξόδου κάθε γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ρ=0,1)

Στα διαγράμματα 8.21 και 8.22 που ακολουθούν βλέπουμε τη primal και dual residual της γεννήτριας 1. Συγκεκριμένα από το διάγραμμα 8.21 βλέπουμε ότι η primal residual $(x_1(k+1) - y_1(k+1))$ έχει μια αρκετά μεγάλη τιμή ($x_1$ αρκετά διαφορετικό του $y_1$) που παραμένει σταθερή μέχρι τη $2^{η}$ (k=1) επανάληψη και στη συνέχεια πέφτει απότομα μέχρι να συναντήσει το 0 (περίπου στην $9^{η}$ επανάληψη). Επίσης από το διάγραμμα 8.22 βλέπουμε ότι η dual residual ( $-\rho \cdot (y_1(k+1) - y_1(k))$) παραμένει σχεδόν μηδενική μέχρι τη $2^{η}$ επανάληψη (k=1) στη συνέχεια κάνει ένα δόντι μέχρι το 0.35 και από την $11^{η}$ επανάληψη και έχει σχεδόν προσεγγίσει το 0. Πάντως το σημαντικό είναι ότι και οι δύο μεταβλητές της γεννήτριας 1 προσεγγίζουν το 0 , άρα πράγματι ο αλγόριθμος συγκλίνει (η primal και dual residual όλων των γεννητριών προσεγγίζουν το 0 απλά ενδεικτικά παρουσιάζουμε μόνος τις μεταβλητές της γεννήτριας 1).

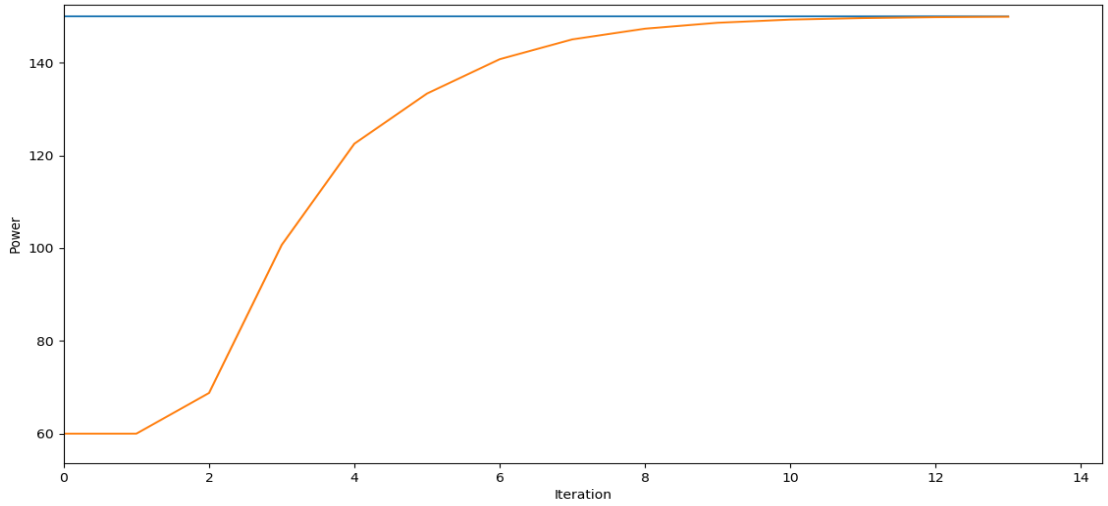



Διάγραμμα 8.21: Η εξέλιξη της primal residual της γεννήτριας 1 (για σύστημα 6 γεννητριών και ρ=0,1)




Διάγραμμα 8.22: Η εξέλιξη της dual residual της γεννήτριας 1 για ρ=0,1 (για σύστημα 6 γεννητριών και ρ=0,1)

Στο διάγραμμα 8.23 μπορούμε να παρατηρήσουμε πως στην αρχή του αλγορίθμου η συνολική παραγωγή των γεννητριών είναι πολύ μικρότερη του φορτίου αλλά σταδιακά αυξάνεται μέχρι ότου στην $11^{η}$ επανάληψη και μετά το φορτίο ικανοποιείται.

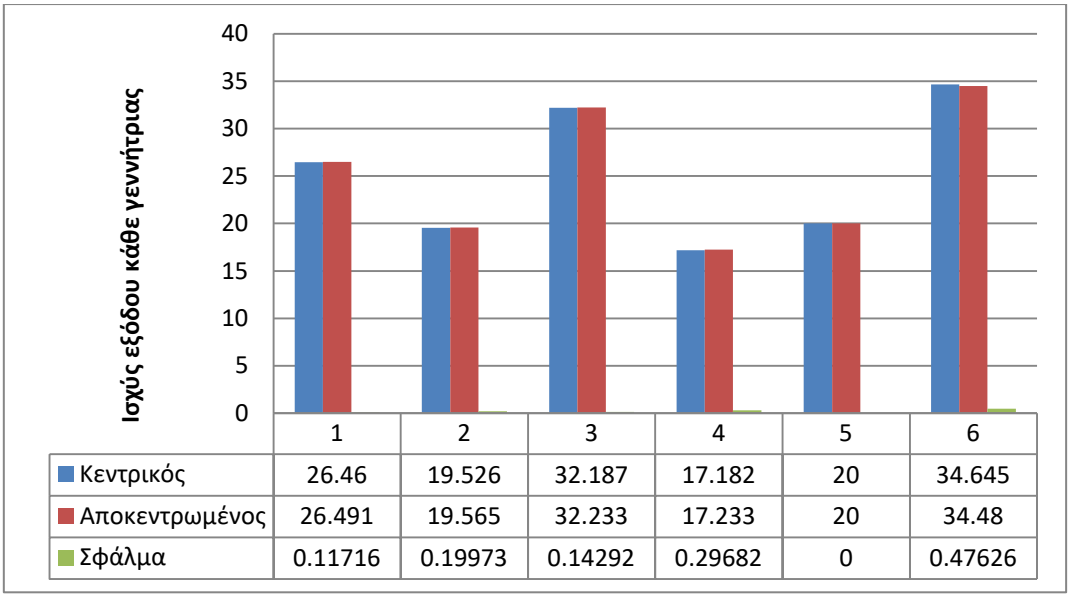



Διάγραμμα 8.23: Ισορροπία παραγωγής-ζήτησης κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ρ=0,1)

Ορίζοντας τη παράμετρο $\rho$ σε 0,1 η μέση κατανάλωση gas μεταβάλλεται από σε 2.237.814.67 που αντιστοιχεί σε μέση χρηματική κατανάλωση $6{,}71 \cdot 10^{-3}$ ether (ή 1,13 €) .

**$\boldsymbol{\rho = 5}$**

Τώρα αυξάνουμε το $\rho$ πάνω από τη μονάδα. Εκτελούμε τον αποκεντρωμένο αλγόριθμο και καταλήγουμε σε σύγκλιση μετά από 259 επαναλήψεις. Στο διάγραμμα 8.24 που ακολουθεί μπορούμε να δούμε την κατανομή ισχύος στις γεννήτριες αλλά και μια σύγκριση με το βέλτιστο κεντρικό αλγόριθμο ώστε να έχουμε μία εκτίμηση του σφάλματος.

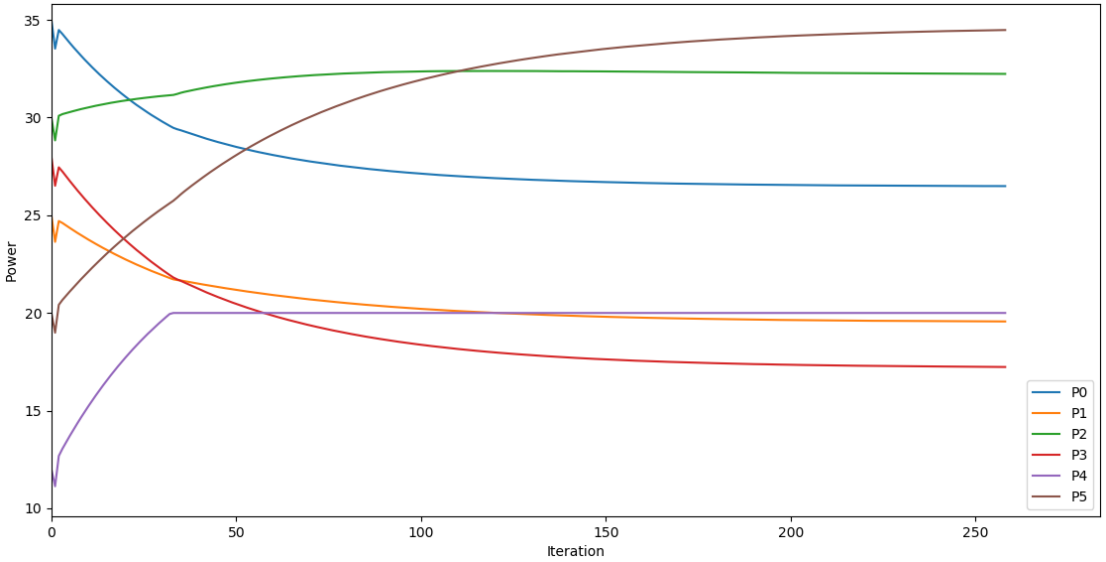



| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Κεντρικός | 26.46 | 19.526 | 32.187 | 17.182 | 20 | 34.645 |
| Αποκεντρωμένος | 26.491 | 19.565 | 32.233 | 17.233 | 20 | 34.48 |
| Σφάλμα | 0.11716 | 0.19973 | 0.14292 | 0.29682 | 0 | 0.47626 |

Διάγραμμα 8.24: Σύγκριση αποκεντρωμένου αλγορίθμου με τον κεντρικό αλγόριθμο (για σύστημα 6 γεννητριών και ρ=5)

Παρατηρώντας τη κατανομή ισχύος που προκύπτει από τον αποκεντρωμένο αλγόριθμο για $\rho = 1$ συμπεραίνουμε ότι δεν διαφέρει πολύ σε σχέση με αυτή που προκύπτει από τον κεντρικό αλγόριθμο( μέσο σφάλμα 0.205%).

Στο διάγραμμα 8.25 που ακολουθεί μπορούμε να δούμε την εξέλιξη της ισχύος εξόδου κάθε γεννήτριας μέχρις ότου λάβει την τελική-σταθερή τιμή της. Επίσης τονίζεται ότι η primal και dual residual κάθε γεννήτριας προσεγγίζει το 0.

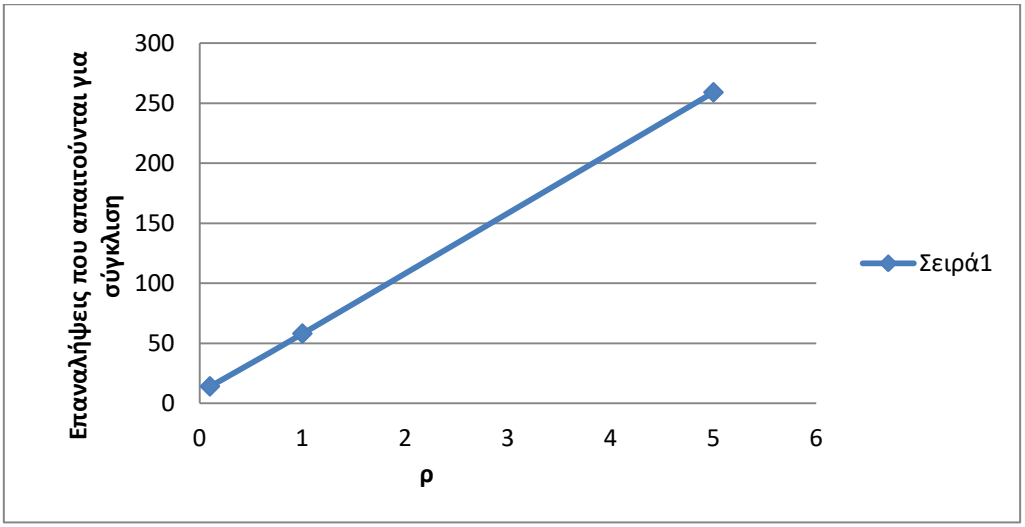



Διάγραμμα 8.25: Η εξέλιξη της ισχύος εξόδου κάθε γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 6 γεννητριών και ρ=5)

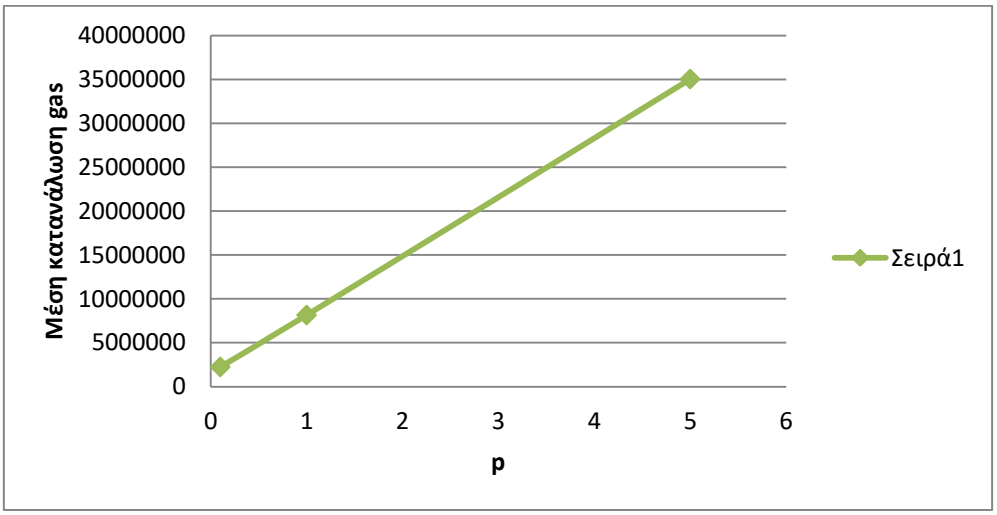
Τέλος αναφέρουμε ότι η μέση κατανάλωση gas είναι 35.033.303,17 που αντιστοιχεί σε μέση χρηματική κατανάλωση 0,105 ether ( ή 17,72 €).

### 8.3.6 Αποτελέσματα του αποκεντρωμένου αλγορίθμου με διαφορετικά ρ

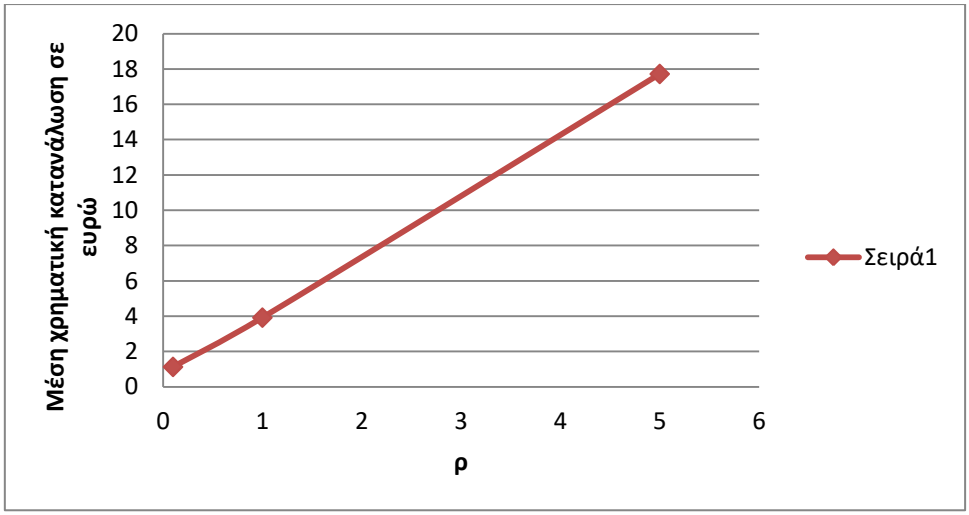
Στα διαγράμματα 8.26, 8.27, 8.28 και 8.29 βλέπουμε πως αλλάζει με τη μεταβολή του ρ, ο αριθμός επαναλήψεων του αλγορίθμου, η μέση κατανάλωση gas και χρημάτων καθώς και το μέσο σφάλμα από τον κεντρικό αλγόριθμο αντίστοιχα.



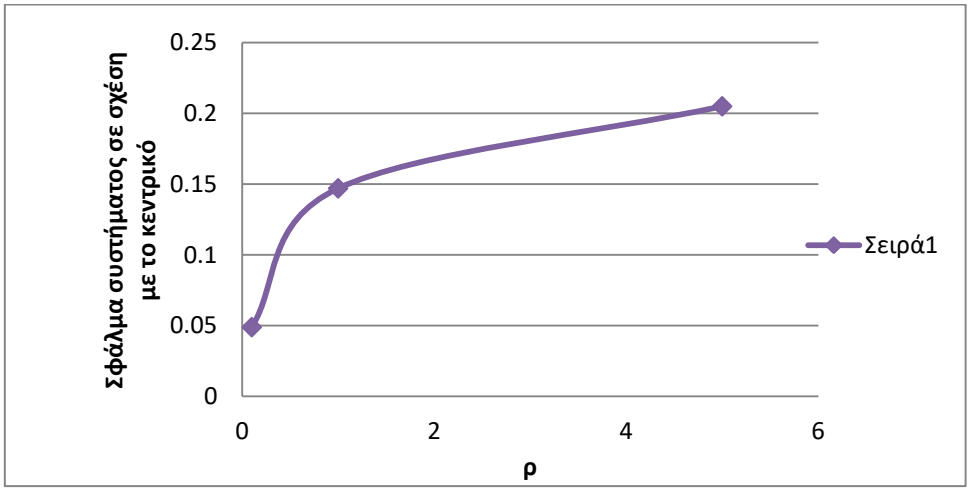

Διάγραμμα 8.26: Αριθμός επαναλήψεων του αποκεντρωμένου αλγορίθμου συναρτήσει του *ρ*




Διάγραμμα 8.27 : Μέση κατανάλωση gas συναρτήσει του *ρ*




Διάγραμμα 8.28: Μέση χρηματική κατανάλωση σε € συναρτήσει του ρ



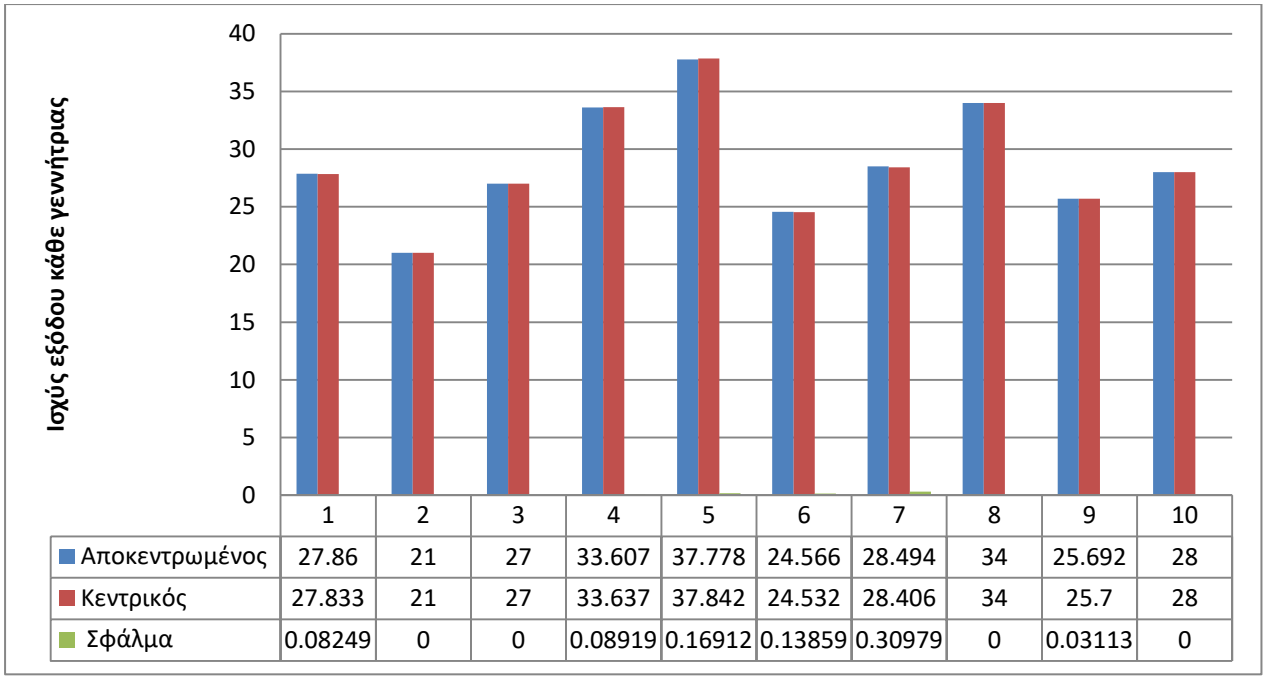

Διάγραμμα 8.29 : Μέσο σφάλμα συναρτήσει του $\rho$

Από τα παραπάνω διαγράμματα γίνεται εύκολα αντιληπτό ότι καθώς το $\rho$ αυξάνεται ο αριθμός επαναλήψεων που απαιτούνται για σύγκλιση καθώς και η μέση κατανάλωση gas – χρημάτων αυξάνονται με γραμμική σχέση . Επίσης το σφάλμα του αποκεντρωμένου συστήματος σε σχέση με το κεντρικό αυξάνεται με την αύξηση του $\rho$.

## 8.4 Συστήματα περισσότερων γεννητριών

### 8.4.1 Σύστημα 10 γεννητριών

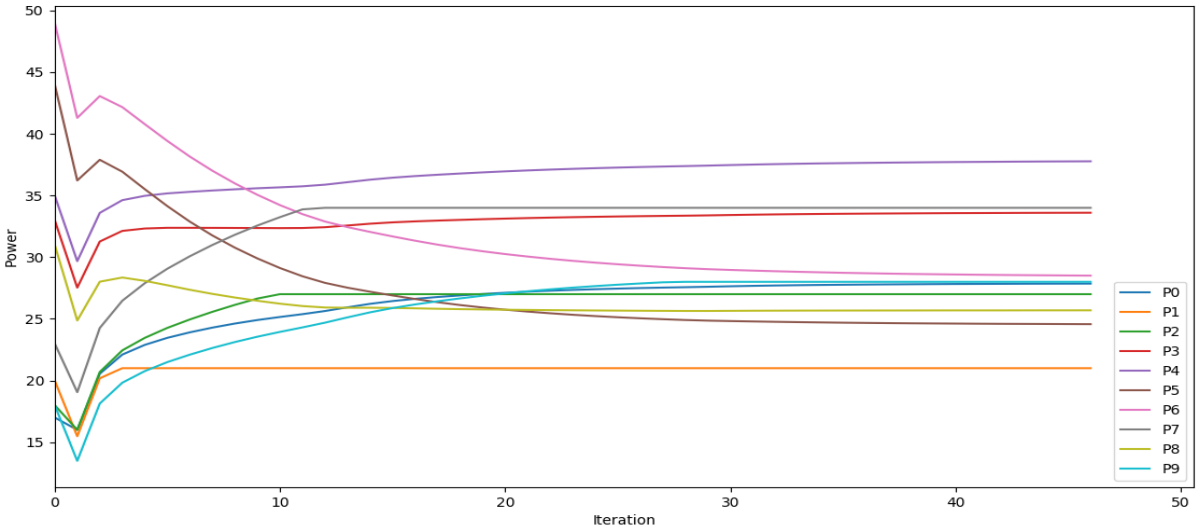
Εκτελώντας τον αποκεντρωμένο αλγόριθμο με είσοδο πλέον 10 γεννήτριες αντί για 6 και φορτίο $P_d = 288kW$ και ε=10 φτάνουμε σε σύγκλιση μετά από 47 επαναλήψεις. Η ενεργός ισχύς εξόδου κάθε γεννήτριας μετά το πέρας του αλγορίθμου καθώς και η σύγκριση του με το αντίστοιχο κεντρικό μοντέλο 10 γεννητριών απεικονίζονται στο διάγραμμα 8.29 που ακολουθεί :




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Αποκεντρωμένος | 27.86 | 21 | 27 | 33.607 | 37.778 | 24.566 | 28.494 | 34 | 25.692 | 28 |
| Κεντρικός | 27.833 | 21 | 27 | 33.637 | 37.842 | 24.532 | 28.406 | 34 | 25.7 | 28 |
| Σφάλμα | 0.08249 | 0 | 0 | 0.08919 | 0.16912 | 0.13859 | 0.30979 | 0 | 0.03113 | 0 |

Διάγραμμα 8.30: Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου για σύστημα 10 γεννητριών

Επίσης από το διάγραμμα 8.30 που ακολουθεί βλέπουμε την εξέλιξη της ισχύος κάθε γεννήτριας κατά τη διάρκεια εκτέλεσης του αλγορίθμου και παρατηρούμε ότι μετά τη $40^{η}$ επανάληψη όλες οι τιμές έχουν περίπου συγκλίνει:

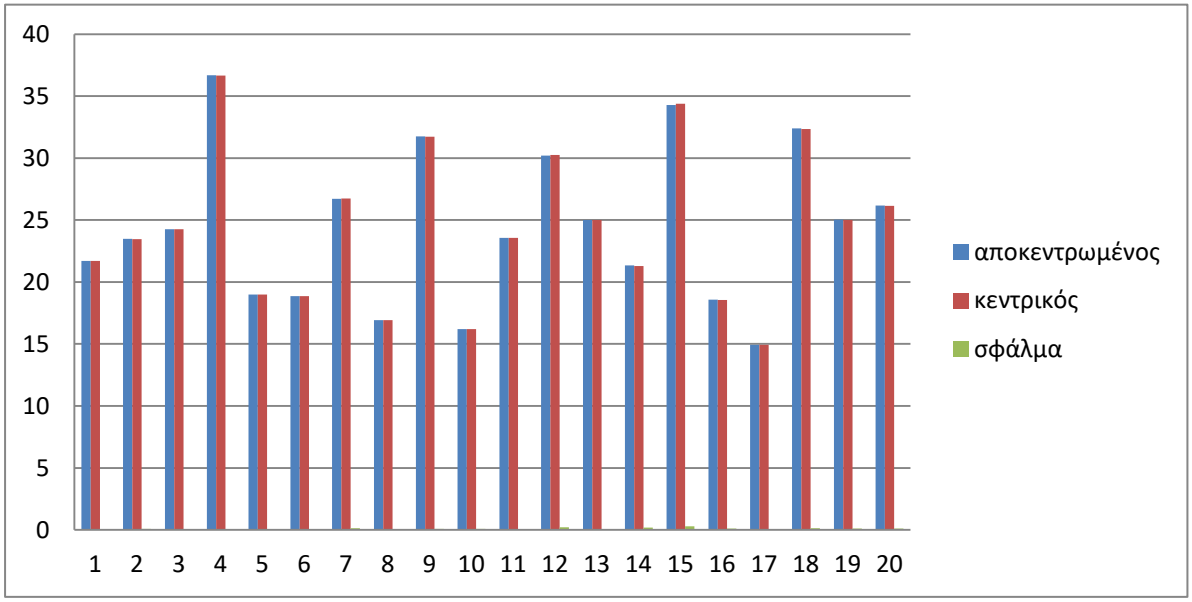



Διάγραμμα 8.31: Η εξέλιξη της ισχύος κάθε γεννήτριας κατά τη διάρκεια του αποκεντρωμένου αλγορίθμου (για σύστημα 10 γεννητριών)

Επίσης η μέση κατανάλωση gas είναι 6.568.191,2 που αντιστοιχεί σε μέση κατανάλωση 0,0197ether (3,32 €).

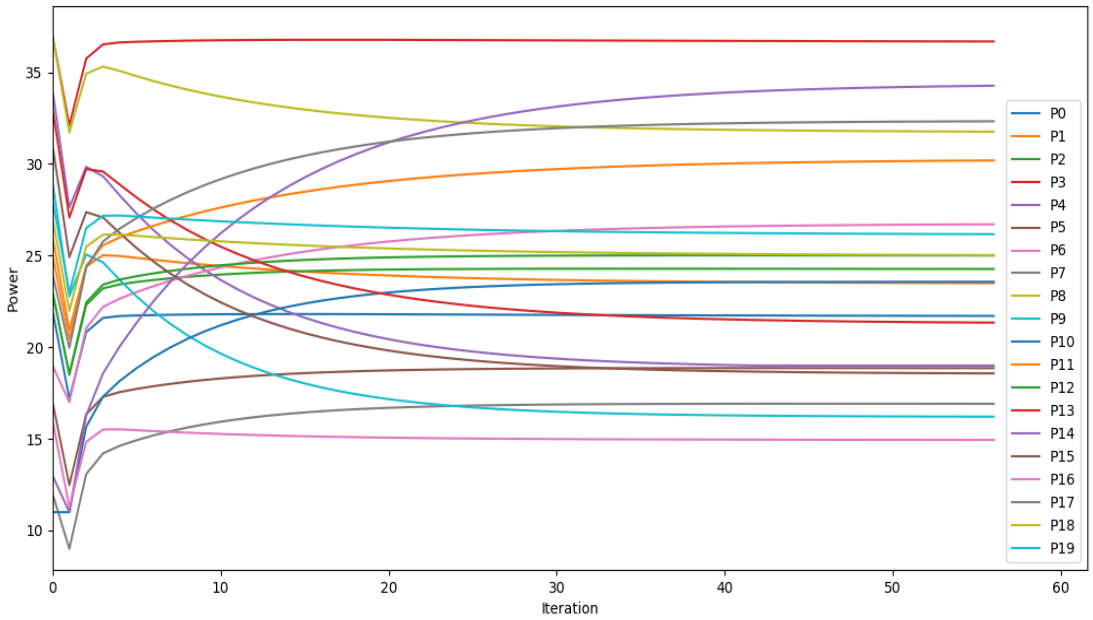
### 8.4.2 Σύστημα 20 γεννητριών

Εκτελώντας τον αποκεντρωμένο αλγόριθμο με είσοδο πλέον 20 γεννήτριες και φορτίο $P_d = 487kW$ και ε=10 φτάνουμε σε σύγκλιση μετά από 57 επαναλήψεις. Στο διάγραμμα 8.31 που ακολουθεί μπορούμε να δούμε μια σύγκριση των τιμών ισχύος που προκύπτουν για σύστημα 20 γεννητριών με τον κεντρικό και με τον αποκεντρωμένο αλγόριθμο:




Διάγραμμα 8.32: Σύγκριση κεντρικού-αποκεντρωμένου αλγορίθμου για σύστημα 20 γεννητριών

Στο διάγραμμα 8.32 βλέπουμε πως η ισχύς εξόδου κάθε γεννήτριας μεταβάλλεται κατά τη διάρκεια του αλγορίθμου μέχρι να πάρει την τελική βέλτιστη τιμή της.

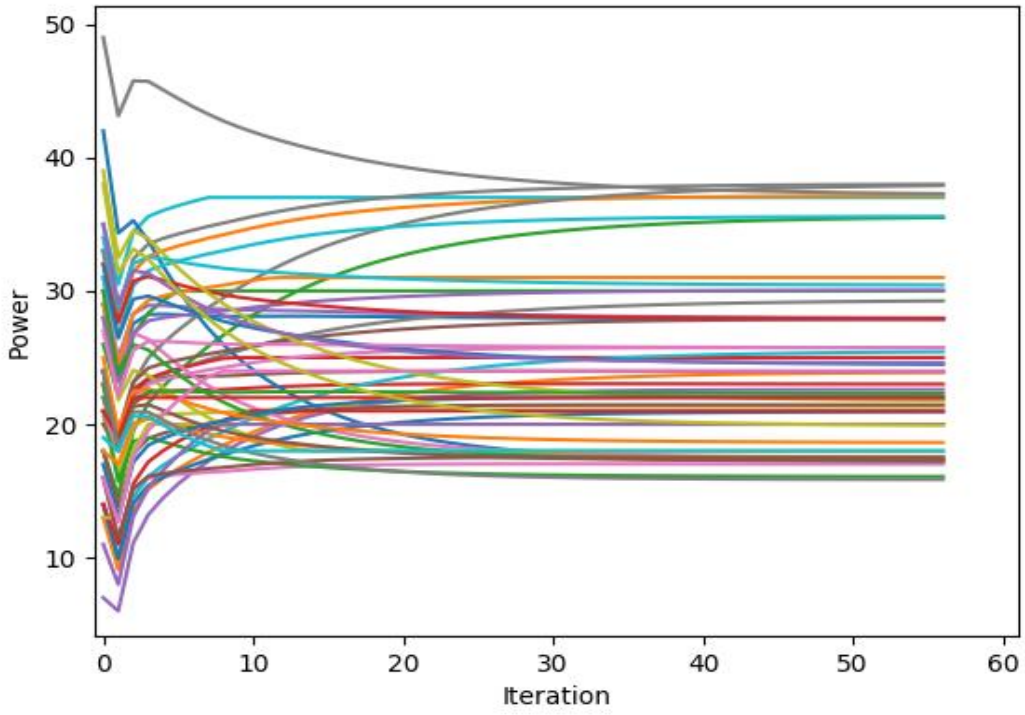



Διάγραμμα 8.33: Εξέλιξη ισχύος εξόδου κάθε γεννήτριας για σύστημα 20 γεννητριών

Επίσης η μέση κατανάλωση gas είναι 8.094.723.2 που αντιστοιχεί σε μέση κατανάλωση 0.0243 ether (4.09 €).

### 8.4.3 Σύστημα 50 γεννητριών

Εκτελώντας τον αποκεντρωμένο αλγόριθμο με 50 γεννήτριες και φορτίο $P_d = 1230kW$ και ε=10 φτάνουμε σε σύγκλιση μετά από 57 επαναλήψεις. Στο διάγραμμα 8.33 βλέπουμε πως η ισχύς εξόδου κάθε γεννήτριας μεταβάλλεται κατά τη διάρκεια του αλγορίθμου μέχρι να πάρει την τελική βέλτιστη τιμή της.

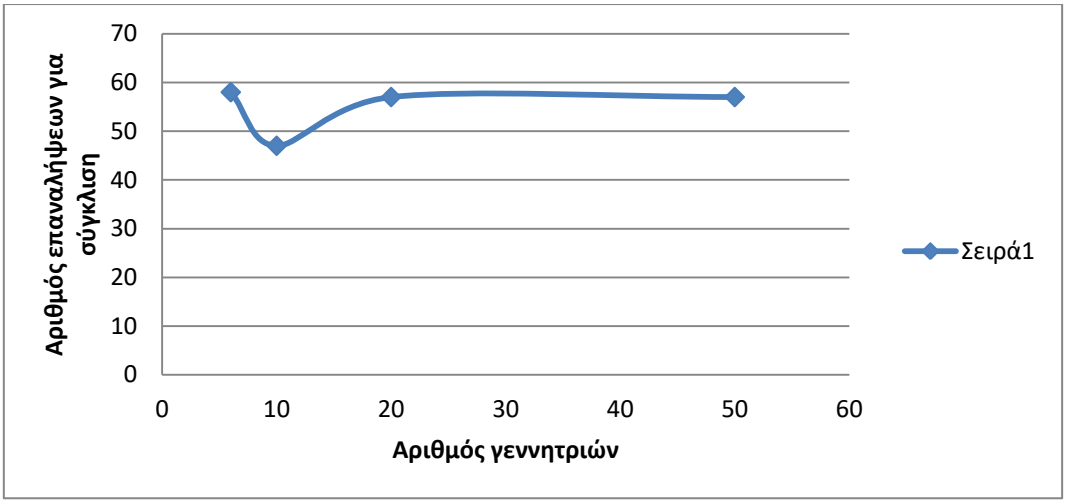



Διάγραμμα 8.34: Εξέλιξη ισχύος εξόδου κάθε γεννήτριας για σύστημα 50 γεννητριών

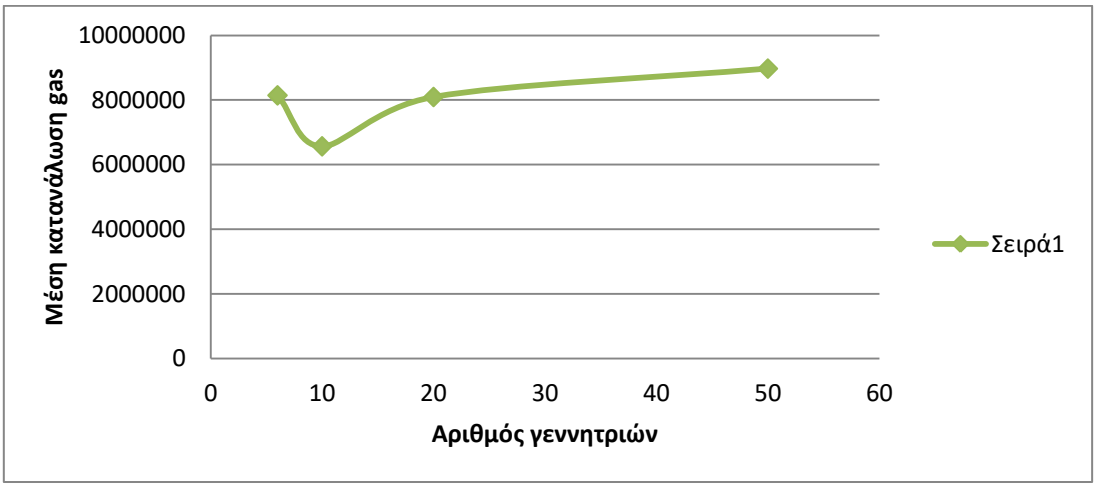
Επίσης η μέση κατανάλωση gas είναι 8.972.855,22 που αντιστοιχεί σε μέση κατανάλωση 0,0269 ether (4,53 €).

### 8.4.4 Αποτελέσματα του αποκεντρωμένου αλγορίθμου για διαφορετικό πλήθος γεννητριών

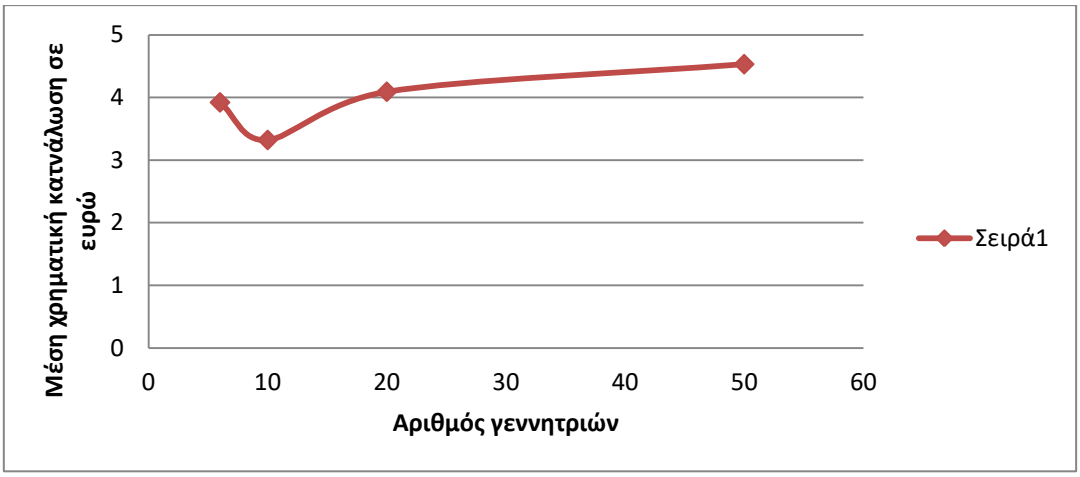
Στα διαγράμματα 8.34 και 8.35 και 8.36, βλέπουμε πως μεταβάλλεται ο αριθμός επαναλήψεων του αλγορίθμου που απαιτούνται για σύγκλιση αλλά και η μέση κατανάλωση gas και € με την αύξηση του αριθμού των γεννητριών :




Διάγραμμα 8.35: Αριθμός επαναλήψεων που απαιτούνται για σύγκλιση συναρτήσει του αριθμού των γεννητριών




Διάγραμμα 8.36: Μέση κατανάλωση gas συναρτήσει του αριθμού των γεννητριών




Διάγραμμα 8.37: Μέση χρηματική κατανάλωση (€) συναρτήσει του αριθμού των γεννητριών

Παρατηρούμε ότι με την αύξηση του αριθμού των γεννητριών από 6 σε 10 ο αριθμός των επαναλήψεων που απαιτούνται για σύγκλιση μειώνεται κατά 11. Αλλά όταν αυξάνουμε τον αριθμό των γεννητριών σε 20 και 50 ο αριθμός των επαναλήψεων που απαιτούνται για σύγκλιση είναι 57 και στις δύο περιπτώσεις, δηλαδή απαιτείται μια λιγότερη επανάληψη για σύγκλιση σε σχέση με το σύστημα των 6 γεννητριών. Όσον αφορά την κατανάλωση gas και χρημάτων, για 10 γεννήτριες έχουμε τη μικρότερη μέση κατανάλωση gas και χρημάτων ενώ για 20 και 50 γεννήτριες η μέση κατανάλωση gas και χρημάτων είναι αρκετά κοντινές και λίγο υψηλότερες από την αντίστοιχες καταναλώσεις για σύστημα 6 γεννητριών.

# Κεφάλαιο 9

## *Συμπεράσματα και προοπτικές*

## 9.1 Συμπεράσματα

Στην παρούσα διπλωματική παρουσιάστηκαν αλγόριθμοι για την επίλυση του προβλήματος Οικονομικής Κατανομής Φορτίου με τη χρήση της μεθόδου ADMM. Αρχικά παρουσιάστηκε εκτενώς η ανάλυση του αλγορίθμου που χρησιμοποιείται στο [72] για τη λύση του προβλήματος Οικονομικής Κατανομής Φορτίου με τη μέθοδο ADMM. Άλλα ο συγκεκριμένος αλγόριθμος είναι κεντρικός γιατί όπως είπαμε για τον υπολογισμό της μεταβλητής $y_i(k+1)$ απαιτείται καθολική πληροφορία. Έτσι λοιπόν προτείναμε τη χρήση του blockchain το οποίο καλείται να υπολογίσει αποκεντρωμένα το κεντρικό βήμα της μεθόδου (υπολογισμός $y_i(k+1)$ καθώς και την ανανέωση των πολλαπλασιαστών Lagrange $\lambda_i(k+1)$ ) καθώς και να ελέγχει αν έχει επιτευχθεί σύγκλιση. Συγκεκριμένα αυτές οι διαδικασίες θα υλοποιούνται από τον κώδικα ενός έξυπνου συμβολαίου το οποίο θα είναι ανεπτυγμένο στο blokchain. Ο κώδικας του συμβολαίου, που θα εκτελείται στην εικονική μηχανή του Ethereum (πρακτικά ο κώδικας θα εκτελείται ταυτόχρονα σε όλους τους πλήρεις κόμβους του δικτύου blockchain), θα συλλέγει από όλους τους κόμβους τις τοπικές λύσεις τους και στη συνέχεια θα υπολογίζονται με βάση τις τιμές ισχύος που έλαβε, οι τιμές ισχύος της κάθε γεννήτριας ώστε να ικανοποιείται το συνολικό φορτίο. Επίσης, θα ανανεώνονται οι πολλαπλασιαστές Lagrange και θα ελέγχεται αν υπάρχει σύγκλιση ώστε να ενημερωθούν οι κόμβοι και να λήξει το πρόβλημα. Επειδή ο κώδικας είναι κοινός για όλους τους κόμβους και εκτελείται ταυτόχρονα σε όλους τους κόμβους, τα αποτελέσματα του αλγορίθμου θα είναι κοινά για κάθε κόμβο. Με αυτό τον τρόπο απαλείφεται η ανάγκη για έναν κεντρικό διαχειριστή που θα εκτελεί τους κεντρικούς υπολογισμούς, ούτε απαιτείται οι κόμβοι να ανταλλάσσουν δεδομένα μεταξύ τους. Τα δεδομένα από κάθε κόμβο αποθηκεύονται στο έξυπνο συμβόλαιο το όποιο εκτελείται με τον ίδιο τρόπο σε κάθε κόμβο (του δικτύου blockchain) και παράγει τα ίδια αποτελέσματα.

Σε επόμενο στάδιο εκτελέστηκαν δοκιμές των αλγορίθμων ώστε να αξιολογηθεί η αποτελεσματικότητά τους. Αρχικά για σύστημα 6 γεννητριών εκτελέστηκε ο κεντρικός αλγόριθμος με $\varepsilon$ σύγκλισης $10^{-14}$ έτσι ώστε να βρεθούν τα βέλτιστα αποτελέσματα του αλγορίθμου. Όπως είδαμε ο αλγόριθμος συγκλίνει μετά από 381 επαναλήψεις και πράγματι εξετάζοντας την primal και dual residual όλων των μεταβλητών παρατηρούμε ότι τείνουν στο 0 και επίσης η ισχύς εξόδου κάθε γεννήτριας είναι εντός των ορίων παραγωγής της και το συνολικό φορτίου του συστήματος ικανοποιείται .

Στη συνέχεια εκτελέσαμε τον αποκεντρωμένο αλγόριθμο για το παραπάνω σύστημα γεννητριών. Αφού εξηγήθηκε ότι η solidity δεν υποστηρίζει δεκαδικούς αριθμούς και ότι τα δεδομένα οι κόμβοι τα στέλνουν πολλαπλασιασμένα επί 1000 ώστε να έχουν ακρίβεια 3 δεκαδικών ψηφίων ορίσαμε το ε=0 και είδαμε ότι επιτυγχάνεται σύγκλιση μετά από 96 επαναλήψεις. Έχουμε σύγκλιση για σε λιγότερες επαναλήψεις σε σχέση με τον κεντρικό αλγόριθμο μιας και στον αποκεντρωμένο αλγόριθμο κρατάμε ακρίβεια 3 δεκαδικών σε σχέση με τον κεντρικό που κρατάμε ακρίβεια 14 δεκαδικών. Πάντως και στο συγκεκριμένο αλγόριθμο παρατηρήσαμε ότι το primal και dual residual όλων των γεννητριών έτεινε στο 0. Τέλος συγκρίναμε τα αποτελέσματα του

αλγορίθμου με τον κεντρικό αλγόριθμο και είδαμε ότι τα σφάλματα είναι αρκετά μικρά (υπάρχει ένα μέσο σφάλμα 0,00735%). Άρα μπορούμε να ισχυριστούμε με ασφάλεια ότι ο αλγόριθμος που τρέχει αποκεντρωμένα λειτουργεί ικανοποιητικά.

Στη συνέχεια ασχοληθήκαμε με την κατανάλωση gas και χρημάτων του κάθε κόμβου. Είδαμε ότι στην περίπτωση όπου το gas price είναι προεπιλεγμένο από το ganache στην τιμή των 20 gWei/gas το κόστος είναι περίπου 44,44 ευρώ. Το κόστος αυτό θεωρείται απαγορευτικό. Βέβαια όπως εξηγήσαμε το gas price συνήθως είναι 3 gWei/gas για τέτοιες εφαρμογές οπότε εκτελέσαμε ξανά τον αποκεντρωμένο αλγόριθμο και είδαμε ότι το μέσο κόστος πέφτει στα 6,67€. Άρα το σύστημα γίνεται πολύ πιο βιώσιμο αλλά και πάλι είναι αρκετά υψηλό το μέσο κόστος. Προκειμένου λοιπόν να μειώσουμε κι άλλο το μέσο κόστος αρχίσαμε να μεταβάλλουμε κάποιες παραμέτρους του αλγόριθμου για να δούμε πως αυτό συμπεριφέρεται.

Αρχικά μεταβάλλαμε το ε σύγκλισης. Πρώτα το αυξήσαμε σε 0,01 και παρατηρήσαμε ότι η σύγκλιση επετεύχθη μετά από 58 επαναλήψεις. Η μείωση των επαναλήψεων που απαιτούνται για σύγκλιση ήταν αναμενόμενο αποτέλεσμα μιας και πλέον οι συνθήκες για σύγκλιση είναι πολύ πιο "χαλαρές". Η ακρίβεια είναι καλή μιας και σε σχέση με τα αποτελέσματα του κεντρικού αλγορίθμου υπάρχει ένα μέσο σφάλμα 0,148%. Όσον αφορά τη μέση κατανάλωση gas κάθε κόμβου αυτή είναι περίπου 8.145.332,17 που αντιστοιχεί σε 0,0232 ether (3,92 €). Αντίστοιχα εκτελέσαμε και τον αλγόριθμο με $\varepsilon$ σύγκλισης ίσο με 20, 50 και 100. Προφανώς με την αύξηση του $\varepsilon$ ο αριθμός των επαναλήψεων που απαιτούνται για σύγκλιση θα μειώνεται με αποτέλεσμα να μειώνεται και η κατανάλωση gas κάθε κόμβου, άρα και χρημάτων (για ε=0,02 η μέση χρηματική κατανάλωση ήταν 3,64 €, για ε=0,05 ήταν 2,90 € για ε=0,1 ήταν 2,36 €) αλλά θα χάνουμε σε ακρίβεια στην εύρεση της ισχύος εξόδου κάθε γεννήτριας. Μια καλή επιλογή είναι το ε να κυμαίνεται από 0,01 έως 0,02 ώστε και να έχουμε πολύ καλή ακρίβεια στα αποτελέσματα μας αλλά και να μην πληρώνει ένας κόμβος τόσα πολλά σε gas όσο με την περίπτωση, όπου ε=0.

Στη συνέχεια μεταβάλλαμε την παράμετρο $\rho$ για να δούμε πως συμπεριφέρεται ο αλγόριθμος. Αρχικά ορίσαμε το $\varepsilon$ σύγκλισης ίσο με 0,01 και το $\rho = 0{,}1$. Παρατηρήσαμε ότι ο αριθμός των επαναλήψεων που απαιτείται για σύγκλιση μειώνεται δραματικά. Πλέον έχουμε σύγκλιση μετά από 14 μόλις επαναλήψεις. Επίσης η ακρίβεια ήταν πάρα πολύ καλή με ένα μέσο σφάλμα τιμών ισχύος σε σχέση με τις αντίστοιχες του κεντρικού αλγόριθμου ίσο με 0,0489%. Όσον αφορά την κατανάλωση gas κάθε κόμβου, μειώθηκε σε 2.237.814.67 που αντιστοιχεί σε $6{,}71 \cdot 10^{-3}$ ether (1,13€). Στη συνέχεια αυξήσαμε το $\rho$ πάνω από τη μονάδα. Συγκεκριμένα με $\rho$=5 και $\varepsilon$=0,01 εκτελέσαμε τον αποκεντρωμένο αλγόριθμο και καταλήξαμε σε σύγκλιση μετά από 259 επαναλήψεις. Όσον αφορά την ακρίβεια το μέσο σφάλμα σε αυτή την περίπτωση ήταν 0,205% και η μέση κατανάλωση gas αυξήθηκε σε 35.033.303,17 που αντιστοιχεί σε 0,105 ether (17,72€). Συνεπώς με την αύξηση του $\rho$ πάνω από τη μονάδα, ο αλγόριθμος αργεί να συγκλίνει με αποτέλεσμα η οικονομική επιβάρυνση των κόμβων να είναι μεγαλύτερη άλλα χάνουμε και σε ακρίβεια. Οπότε η χρήση $\rho > 1$ μάλλον δεν ενδείκνυται. Επίσης, το γεγονός ότι το έξυπνο συμβόλαιο που έχει αναπτυχθεί δεν

προβλέπει τη χρήση δεκαδικών αριθμών $\rho > 1$ μάλλον δεν είναι τόσο αρνητικό. Όλα τα παραπάνω αποτελέσματα οδηγούν στο συμπέρασμα ότι το πρόβλημα Οικονομικής Κατανομής Φορτίου προκειμένου να είναι και ακριβές όσον αφορά την κατανομή ισχύος στις γεννήτριες αλλά και οικονομικά βιώσιμο θα πρέπει να επιλέγονται τιμές του $0 < \rho \leq 1$.

Τέλος ο αποκεντρωμένος αλγόριθμος εκτελέστηκε και για δίκτυα περισσότερων γεννητριών. Συγκριμένα εκτελέστηκε για δίκτυο 10, 20 και 50 γεννητριών. Για 10 γεννήτριες χρειάστηκαν 46 επαναλήψεις για σύγκλιση, ενώ για 20 και 50 χρειάστηκαν 57. Άρα φαίνεται πως η αύξηση του αριθμού των γεννητριών δεν αυξάνει σημαντικά με την αύξηση των κόμβων. Επίσης η κατανάλωση gas ήταν αρκετά όμοια για 20 και 50 γεννήτριες και λίγο μεγαλύτερη από την κατανάλωση gas για σύστημα 6 γεννητριών (παρότι ο αριθμός των γεννητριών ήταν πολύ μεγαλύτερος) ενώ στην περίπτωση των 10 γεννητριών ήταν μικρότερη σε σχέση με το σύστημα 6 γεννητριών. Άρα μάλλον ο αριθμός των επαναλήψεων που απαιτούνται για σύγκλιση και επομένως και η μέση κατανάλωση gas/χρημάτων φαίνεται πως δεν εξαρτάται από το πλήθος των γεννητριών αλλά από τις παραμέτρους τους.

## 9.2 Προοπτικές για περαιτέρω έρευνα

Στο πλαίσιο αυτής της εργασίας είδαμε ότι κάθε φορά ο κόμβος που στέλνει στο έξυπνο συμβόλαιο τελευταίος τη λύση του $x_i(k+1)$ (εδώ είναι πάντα η τελευταία διεύθυνση της whitelist) χρεώνεται όλα τα υπολογιστικά βήματα τα οποία εκτελεί η εικονική μηχανή του Ethereum για τον υπολογισμό των $y_i(k+1),\ \lambda_i(k+1)$ και τον έλεγχο για σύγκλιση. Έτσι πληρώνει τα περισσότερα χρήματα από όλους. Έτσι λοιπόν θα μπορούσε να ερευνηθεί ένας τρόπος ώστε το κόστος αυτών των υπολογιστικών βημάτων να κατανέμεται ισομερώς σε κάθε κόμβο.

Ένα άλλο βήμα που θα μπορούσε να γίνει για ώστε ο κώδικας ακόμα πιο αποδοτικός είναι να βρεθεί τρόπος οι πράξεις στην solidity να γίνονται με μεγαλύτερη ακρίβεια. Επιπλέον το ζήτημα της εισαγωγής της παραμέτρου $\rho$ με περισσότερα δεκαδικά ψηφία και όχι μόνο τιμών $\rho$ που το $1/\rho$ είναι ακέραιος αριθμός (ιδιαίτερα για $0 < \rho \leq 1$) θα ήταν μια καλή επέκταση.

Τέλος το πρόβλημα θα μπορούσε να επεκταθεί περεταίρω και να γίνει πιο ρεαλιστικό με την θεώρηση των απωλειών των γραμμών μεταφοράς καθώς και την χρησιμοποίηση μονάδων όπως υδροηλεκτρικά ή σταθμοί συμπαραγωγής που έχουν διαφορετικές συναρτήσεις κόστους.

## Βιβλιογραφία


[1]Κ. Βουρνάς, Γ. Κονταξής, "Εισαγωγή στα Συστήματα Ηλεκτρικής Ενέργειας" Εκδόσεις Συμμετρία, Αθήνα 2010

[2] Centralized Generation of Electricity and its Impacts on the Environment, Available at: https://www.epa.gov/energy/centralized-generation-electricity-and-its-impacts-environment Πρόσβαση 04/09/2019

[3]Distributed energy: Driving the ghosts out of the machine, Available at: https://grist.org/climate-energy/distributed-energy-driving-the-ghosts-out-of-the-machine/ , Πρόσβαση 04/09/2019

[4]Κ. Βουρνάς, Β.Κ. Παπαδιάς, Κ. Ντελκής "Παραγωγή Ηλεκτρικής Ενέργειας, Έλεγχος και Ευστάθεια Συστήματος" Εκδόσεις Συμμετρία, Αθήνα 2011

[5]Εισαγωγή στη Διεσπαρμένη παραγωγή, University of Cyprus, Department of Electrical and Computer Engineering, http://www.eng.ucy.ac.cy/elias/Courses/ECE445/presentations/Lectures2010/Distributed%20Generation1.pdf , Πρόσβαση 04/09/2019

[6] 1 G. Pepermans, J. Driesen, D. Haeseldonckx, R. Belmans, W. D'haeseleer, "Distributed generation: definition, benefits and issues", April 2005.

[7] Α. Τσικαλάκης, Διδακτορική Διατριβή, "Συμβολή στον προγραμματισμό λειτουργίας Δικτύων Διανομής Ηλεκτρικής Ενέργειας με μεγάλη διείσδυση διεσπαρμένης και ανανεώσιμης παραγωγής και συσκευών αποθήκευσης", Εθνικό Μετσόβιο Πολυτεχνείο, Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών, Ιούλιος 2008.

[8] Ευφυή δίκτυα διανομής για αυξημένη διείσδυση διεσπαρμένης παραγωγής, καθ. Νίκος Χατζηαργυρίου, Εκδήλωση ΕΒΕΑ –Εφημερίδας Απογευματινή «ΕΝΕΡΓΕΙΑ – ΦΩΤΟΒΟΛΤΑΙΚΑ ΠΑΡΚΑ»

[9] Σπυρίδων Χ. Αποστολόπουλος "Αποκεντρωμένη ρύθμιση τάσης σε δίκτυα διανομής", Διπλωματική Εργασία , Εθνικό Μετσόβιο Πολυτεχνείο, Αθήνα, Οκτώβριος 2016

[10]Electric Power Research Institute, Available at : https://smartgrid.epri.com, Πρόσβαση 05/09/2019

[11] Ιάσονας, Ν. Κουβελιώτης – Λυσίκατος, "Συμβολή στην ανάπτυξη κατανεμημένων αλγορίθμων για τον έλεγχο και τη λειτουργία ευφυών δικτύων διανομής", Διδακτορική Διατριβή , Εθνικό Μετσόβιο Πολυτεχνείο, Αθήνα, Μάρτιος 2019

[12] Βασίλειος Παπακωνσταντίνου, "Ευφυή Δίκτυα στο Ελληνικό ΣΗΕ", Διπλωματική Εργασία, Πανεπιστήμιο Θεσσαλίας, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Βόλος 2016

[13]Χρήστος Αρ. Νικολόπουλος," Εφαρμογή υβριδικού AC/DC Μικροδικτύου για την ενσωμάτωση Οικιακών Φωτοβολταϊκών Συστημάτων σε Ευφυή Δίκτυα" , Διπλωματική Εργασία, Εθνικό Μετσόβιο Πολυτεχνείο , Αθήνα, Μάρτιος 2015

[14] Claudia Pop, Tudor Cioara, Marcel Antal, Ionut Anghel, Ioan Salomie and Massimo Bertoncini ,"Blockchain Based Decentralized Management of Demand Response Programs in Smart Energy Grids" , 9 January 2018

[15]How electric vehicles help stabilize the electric grid, Availiable at: https://v2g-clarity.com/knowledgebase/vehicle-to-grid/ , Πρόσβαση 06/09/2019

[16] M.A. Matin and M.M. Islam ,"Overview of Wireless Sensor Network"

[17] Stamatis Karnouskos, "The cooperative Internet of Things enabled Smart Grid", SAP Research (www.sap.com), Karlsruhe, Germany

[18] Louise Hagström, Olivia Dahlquist,"Scaling blockchain for the energy sector", Uppsala University, June 2017

[19] Γεώργιος Ν. Παπαδόδημας ,"Ανάπτυξη Έξυπνων Συμβολαίων στο Blockchain και εφαρμογή στο IoT", Διπλωματική Εργασία, ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ, Αθήνα, Μάρτιος 2018

[20] Adam Hahn, Rajveer Singh, Chen-Ching Liu , Sijie Chen "Smart Contract-based Campus Demonstration of Decentralized Transactive Energy Auctions",2017

[21] Taneli Hukkinen, "Reducing blockchain transaction costs in a distributed energy market application", Master's Thesis, Espoo, January 17, 2018

[22] Satoshi Nakamoto ,"Bitcoin: A Peer-to-Peer Electronic Cash System"

[23] "Blockchain Architecture Basics: Components, Structure, Benefits & Creation", Available at: https://mlsdev.com/blog/156-how-to-build-your-own-blockchain architecture, Πρόσβαση: 08/09/2019

[24] Xigao Wu, Bin Duan, Yinxin Yan,Ying Zhong "M2M Blockchain: The Case of Demand Side Management of Smart Grid"

[25]https://www.researchgate.net/figure/A-simplified-example-of-how-blocks-are-chained-to-form-a-blockchain-Notice-that-each_fig1_332215097, Πρόσβαση : 08/09/2019

[26] Patrick Schueffel, Nikolaj Groeneweg, Rico Baldegger, "TheCrypto Encyclopedia", Published in Switzerland, 2019

[27] “Cryptographic hash functions explained: A beginner’s guide”, Available at : https://komodoplatform.com/cryptographic-hash-function/ , Πρόσβαση : 07/09/2019

[28] “Cryptographic primitives”, Available at: https://blockchain-course.org, Πρόσβαση : 07/09/2019

[29] “Cryptography”, Available at: https://lisk.io/academy/blockchain-basics/how-does-blockchain-work/blockchain-cryptography-explained , Πρόσβαση : 08/09/2019

[30] Darrel Hankerson, Alfred Menezes, Scott Vanstone, “Elliptic Curve Cryptography”

[31] Don Johnson, Alfred Menezes, Scott Vanstone, “The Elliptic Curve Digital Signature Algorithm (ECDSA)”,Certicom Research, Canada, Published online: 27 July 2001

[32] Zibin Zheng, Shaoan Xie, Hongning Dai, Xiangping Chen and Huaimin Wang, “An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends”, 2017 IEEE 6th International Congress on Big Data

[33]Michael Crosby, Nachiappan, Pradan Pattanayak, Sanjeev Verma, Vignesh Kalyanaraman, “BlockChain Technology: Beyond Bitcoin”, Applied Innovation Review, Issue No. 2 June 2016

[34]Our choice of digital signature algorithm, Available at: https://exonum.com/blog/09-27-17-digital-signature/, Πρόσβαση: 08/09/2019

[35] “Transactions, UTXO and the Bitcoin Application Layer”, Available at: https://blockchain-course.org, Πρόσβαση : 08/09/2019

[36] “learn me a bitcoin”, Available at: https://learnmeabitcoin.com/glossary/p2pk, Πρόσβαση 08/09/2019

[37]”Cryptocurrency Wallet Guide: A Step-by-Step Tutorial”, Available at: https://blockgeeks.com/guides/cryptocurrency-wallet-guide/ , Πρόσβαση:08/09/2019

[38] Asutosh Palai, Meet Vora, Aashaka Shah,” Empowering Light Nodes in Blockchains with Block Summarization”

[39] Juan Garay, Aggelos Kiayias, and Nikos Leonardos,”The Bitcoin Backbone Protocol: Analysis and Applications”, International Association for Cryptologic Research 2015

[40] E O Kiktenko, N O Pozhar, M N Anufriev , A S Trushechkin , R R Yunusov, Y V Kurochkin, A I Lvovsky, and A K Fedorov,” Quantum-secured blockchain”, 31 May 2018

[41] Cynthia Dwork, Moni Naor, “Pricing via Processing or Combatting Junk Mail”

[42] Markus Jakobsson, Ari Juels, “PROOFS OF WORK AND BREAD PUDDING PROTOCOLS (EXTENDED ABSTRACT)”, Springer Science+Business Media Dordrecht 1999

[43] “What Is a 51% Attack ?”, Available at: https://www.binance.vision/security/what-is-a-51-percent-attack, Πρόσβαση 09/09/2019

[44] “PROOF-OF-WORK IS DRAINING THE GRID” Available at : https://blog.xtrabytes.global/technology/proof-of-work/ , Πρόσβαση : 09/09/2019

[45]Ittay Eyal, Emin Gün Sirer, “Majority Is Not Enough: Bitcoin Mining Is Vulnerable”, Communications of the ACM, July 2018, Vol.61 No.7, Pages 95-102

[46] “Mining Centralization Senarios”, Available at: https://medium.com/@jimmysong/mining-centralization-scenarios-b74102adbd36 , Πρόσβαση: 09/09/2019

[47] Pavel Vasin, “BlackCoin’s Proof-of-Stake Protocol v2”

[48] “Proof of Stake (PoS)”, Available at: https://www.investopedia.com/terms/p/proof-stake-pos.asp , Πρόσβαση: 09/09/2019

[49] Iddo Bentov, Charles Lee, Alex Mizrahi, Meni Rosenfeld , “Proof of Activity: Extending Bitcoin’s Proof of Work via Proof of Stake”,

[50] Aggelos Kiayias, Ioannis Konstantinou, Alexander Russell, “A Provably Secure Proof-of-Stake Blockchain Protocol”, September 12, 2016

[51]BitFury Group, Jeff Garzik “Public versus Private Blockchains Part 1: Permissioned Blockchains ,White Paper”, Oct 20, 2015 (Version 1.0)

[52] Vitalik Buterin, "Ethereum white paper", 2013, Available at: https://github.com/ethereum/wiki/wiki/Ethereum-White-Paper, Πρόσβαση: 10/09/2019.

[53]Dr. Gavin Wood, “Ethereum: A Secure Decentralised Generalised Tranasaction Ledger”, Byzantium Version e7515a3 - 2019-08-16

[54]Ethereum , Available at: https://blockchain-course.org/lecture/ethereum, Πρόσβαση: 11/09/2019

[55] “How does Ethereum work, anyway? ”, Available at: https://medium.com/@preethikasireddy/how-does-ethereum-work-anyway-22d1df506369 , Πρόσβαση: 11/09/2019

[56] Li Jun Wu, Kun Meng, Shuo Xu, Shu Qin Li, Meng Ding, Yan Feng Suo, “Democratic Centralism: a hybrid Blockchain architecture and its applications in Energy Internet”, First IEEE International Conference on Energy Internet

[57] Tianyu Yang, Hongbin Sun, Wenlu Zhao, “Applying Blockchain to Decentralized Operation in Future Energy Internet”, 2017,IEEE

[58] Michael J. Ashley and Mark S. Johnson, “Establishing a Secure, Transparent, and Autonomous Blockchain of Custody for Renewable Energy Credits and Carbon Credits”, 2018 IEEE

[59] Jonas Schlund, Lorenz Ammon and Reinhard German,” ETHome: Open-source blockchain based energy community controller”, e-Energy ’18, June 12–15, 2018, Karlsruhe, Germany

[60] Khaled Shuaib, Juhar Ahmed Abdella, Farag Sallabi and Mohammed Abdel-Hafez, ” Using Blockchains to Secure Distributed Energy Exchange”, 5th International Conference on Control, Decision and Information Technologies, Thessaloniki, Greece, April 10-13, 2018

[61] Yu Nandar Aung, Thitinan Tantidham,” Review of Ethereum: Smart Home Case Study”, 2017 2nd International Conference on Information Technology (INCIT)

[62] M.L. Di Silvestre, P. Gallo, M.G. Ippolito, E. Riva Sanseverino, G. Sciumè and G. Zizzo, “An Energy Blockchain, a use case on Tendermint”, DEIM ,Palermo, Italy

[63]Παπαδιάς, Β., Κονταξής, Γ. (2003). Ηλεκτρική Οικονομία. Αθήνα: Εθνικό Μετσόβιο Πολυτεχνείο.

[64] Όλγας Κάγιαλη.,“Ανάπτυξη Κατανεμημένων Αλγορίθμων για Κατανεμημένη Οικονομική Κατανομή Φορτίου” , Διπλωματική Εργασία, Εθνικό Μετσόβιο Πολυτεχνείο, Αθήνα,2016

[65] Jizhang Zhu, “Optimization of power system operation”

[66]Παντελής Δράτσας, “Αλγόριθμος Αποκεντρωμένης Οικονομικής Κατανομής Φορτίου λαμβάνοντας υπόψη Στοχαστική Αιολική Παραγωγή”, Διπλωματική Εργασία, Εθνικό Μετσόβιο Πολυτεχνείο, Αθήνα,2017

[67]Παύλος Σ. Γεωργιλάκης, “Σύγχρονα Συστήματα Μετάφοράς και Διανομής Ηλεκτρικής Ενέργειας” , Αθήνα, Ιούλιος 2015

[68] Grainger, J., Stevenson, W. and Stevenson, W. (1994). Power system analysis.

[69] Stephen Boyd, Neal Parikh, Eric Chu Borja Peleato and Jonathan Eckstein, “Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers”

[70]Dimitris Christodouleas , “Distributed convex optimization using ADMM”, Technical University of Crete, December 2014

[71] E. Münsing, J. Mather and S. Moura, "Blockchains for decentralized optimization of energy resources in microgrid networks", 2017 IEEE Conference on Control Technology and Applications (CCTA), Mauna Lani, HI, 2017

[72] Peng Li, Jiangping Hu , “An ADMM Based Distributed Finite-Time Algorithm for Economic Dispatch Problems”, May 2018

[73] Solidity documentation Available at: https://solidity.readthedocs.io/en/v0.5.0/, Πρόσβαση 29/08/2019

[74] Διπλωματική Εργασία του Γεώργιου Ν. Παπαδόδημα: Ανάπτυξη Έξυπνων Συμβολαίων στο Blockchain και εφαρμογή στο IoT. Αθήνα.Μάρτιος 2018

[75] Remix documentation Available at: https://remix-ide.readthedocs.io/en/latest/, Πρόσβαση 29/08/2019

[76] Ganache, Available at: https://www.trufflesuite.com/docs/ganache/overview, Πρόσβαση 29/08/2019.

[77]Python, Available at: https://www.python.org/doc/essays/blurb/, Πρόσβαση 29/08/2019

[78]Εισαγωγή στον προγραμματισμό με την python , Νικόλαος Α. Αγγελιδάκης, Α’ Έκδοση ,Ηράκλειο , Αύγουστος 2015

[79]Web 3.py documentation, Available at: https://web3py.readthedocs.io/en/stable/, Πρόσβαση 29/08/2019.

[80] http://www.dappuniversity.com/articles/web3-py-intro ,Πρόσβαση 29/08/2019

[81] Numpy, Available at: https://numpy.org, Πρόσβαση 29/08/2019

[82]Pandas, Available at: https://pandas.pydata.org, Πρόσβαση 29/08/2019

[83]Matplotlib.pyplot, Available at: https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.pyplot.html, Πρόσβαση 29/08/2019

## Παράρτημα

Κώδικας Python που εκτελείται τοπικά σε κάθε κόμβο:

```
import json
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from web3 import Web3

#connection with ganache
web3=Web3(Web3.HTTPProvider('HTTP://127.0.0.1:7545'))
addresses=web3.eth.accounts #load the accounts of the ganache in an array

N=len(addresses)

balances=np.zeros((N))
totalGas=np.zeros((N))
spentEther=np.zeros((N))

#create a contract instance
contract_address="0x3Db6e2407F41EaE4D6f08e63e0d454F21121AC7a"
abi =json.loads('abi')
contract=web3.eth.contract(address=contract_address , abi=abi)
gasprice=3.e+9
#read input-csv
basic_data = pd.read_csv('../Data/input.csv',header=None)
basic_data.columns = ['a','b','c','x_min','x_max','x','l','y','p','Pd']
a=np.array(basic_data['a'])
b = np.array(basic_data['b'])
c = np.array(basic_data['c'])
x_min = np.array(basic_data['x_min'])
x_max = np.array(basic_data['x_max'])
x_= np.array(basic_data['x'])
x = np.vstack((x_,np.zeros((1000,N))))
l_ = np.array(basic_data['l'])
l = np.vstack((l_,np.zeros((1000,N))))
y_ = np.array(basic_data['y'])
y = np.vstack((y_,np.zeros((1000,N))))
p=float(basic_data['p'][0])
Pd=int(basic_data['Pd'][0])

problemSolved = False
mul = 1000; #because solidity doed not support floats
```

```
#init ( send the initial values x[0][i])
k=0
for i in range(N):
 web3.eth.defaultAccount=addresses[i]
 tx_hash=contract.functions.submitInitialValue(int(x[k,i]*mul) ,i).transact()
 web3.eth.waitForTransactionReceipt(tx_hash,timeout=120)

diffx_y = np.zeros((1000))
diffy_y = np.zeros((1000))
#compute x[k+1][i]
while True:
    for i in range (N):

        l[k][i]=contract.functions.l(k,i).call()
        y[k][i]=contract.functions.y(k,i).call()
        l[k][i] = l[k][i]/(mul)
        y[k][i] = y[k][i]/(mul)
        x[k+1,i]=min(max((p*y[k,i]-l[k][i]-b[i])/(2*a[i]+p),x_min[i]),x_max[i])
        web3.eth.defaultAccount=addresses[i]
        tx_hash=contract.functions.submitValue(int(x[k+1,i]*mul) ,i,k).transact()
        web3.eth.waitForTransactionReceipt(tx_hash,timeout=120)

    y[k+1][0]=contract.functions.y(k+1,0).call()
    diffx_y[k] = abs(x[k+1,0]-(y[(k+1),0]/1000))

    if k != 0:
        diffy_y[k-1] = y[(k),0] - y[(k-1),0]
    problemSolved = contract.functions.problemSolved().call()
    if(problemSolved==True):
        print("convergence",k)
        y[k+1][0]=contract.functions.y(k+1,0).call()
        diffy_y[k] = (y[(k+1),0]/1000) - y[(k),0]

        break
    k+=1
#compute the total gas per node and ether tah spent
for i in range (N):
    balances[i]=web3.eth.getBalance(addresses[i])
totalGas=(-balances+100.e+18)/gasprice
spentEther=(-balances+100.e+18)/(1.e+18)
maxGas=np.max(totalGas)
minGas=np.min(totalGas)
averageGas=np.average(totalGas)
averageEther=np.average(spentEther)
```

```
#plots
#evolution of power generation
figure1 = plt.figure(1)
for i in range(N):
    plt.plot(range(0,k+1),x[:k+1,i])

plt.ylabel('Power')
plt.xlabel('Iteration')
names = []
for i in range(0,20):
    names.append( 'P'+str(i))
plt.legend(names,loc='lower right')

#evolution of primal residual of the first generator
figure2 = plt.figure(2)
plt.plot(range(0,k+1),diffx_y[:(k+1)])
plt.ylabel('Primal residual')
plt.xlabel('Iteration')


diffy_y = (-p)*diffy_y

#evolution of dual residual of the first generator

figure3 = plt.figure(3)
plt.plot(range(0,k+1),diffy_y[:(k+1)])
plt.ylabel('Dual residual')
plt.xlabel('Iteration')
#evolution of supply-demad balance

figure4 = plt.figure(4)
sum_x = np.zeros((k+2))

for i in range(k+2):
    for j in range(N):
        sum_x[i]+=x[i,j]

pd_array = np.full((k+1,),Pd)
plt.plot(range(0,(k+1)),pd_array)
plt.plot(range(0,(k+1)),sum_x[1:(k+2)])

plt.ylabel('Power')
plt.xlabel('Iteration')

plt.show()
```

Κώδικας έξυπνου συμβολαίου:

```
pragma solidity ^0.5.0;

contract aggregator{
    uint public N;
    int[1000][1000] public l;
    int [1000][1000] public z;
    int p=1;
    int Pd;
    int [1000] public h;
    int[1000][1000] public x;
    int[1000][1000] public y;
    uint16 public k;
    int epsilon;
    int f_flag;
    address[] public whitelist;
        mapping (address => bool) public waiting;
        bool public problemSolved;
        bool public init;


 constructor (address[] memory _whitelist,uint _N ,int _p, int _Pd, int _epsilon, int _f_flag) public{
     p=_p;
     N=_N;
     Pd=_Pd*1000;
     whitelist=_whitelist;
     resetWaiting();
     problemSolved=false;
     init=true;
     k=0;
     epsilon = _epsilon;
     f_flag = _f_flag;
 }

function submitInitialValue(int initialValue,uint i ) public{
require(!problemSolved);
require(init);


if(waiting[msg.sender])
  {
      if(msg.sender==whitelist[i])
      {
              x[0][i]=initialValue;
              y[0][i]=x[0][i];
              waiting[msg.sender]=false;
      }
else{revert();}
}
else{revert();}

if(!stillWaiting())
   {
     resetWaiting();
     init=false;
   }
}
```

```
function submitValue(int value,uint i, uint16 iteration ) public{
    require(!problemSolved);
    require(iteration==k);
     if(waiting[msg.sender])
     {
          if(msg.sender==whitelist[i])
             {
             x[k+1][i]=value;
             waiting[msg.sender]=false;
             }
      else{revert();}
     }
else{revert();}

if(!stillWaiting())
   {
        if (f_flag == 0) updateY();
        else updateY_float();
        resetWaiting();
        k++;

   }

}

function updateY() public {
    for(uint i=0; i<N; i++){
        // compute y[k+1][i], l[k+1][i] if p int

        if(i==0) z[k][i]=x[k+1][i]+(l[k][i])/p;
        else z[k][i]=z[k][i-1]+x[k+1][i]+(l[k][i])/p;

    }

    h[k]=(p*((Pd)-z[k][N-1]))/int(N);

    for(uint i; i<N; i++){

      y[k+1][i]=(h[k]/p)+x[k+1][i]+(l[k][i])/p;
      l[k+1][i]=l[k][i]+p*(x[k+1][i]-y[k+1][i]);
    }

    //check for convergence


    for(uint i=0; i<N; i++){
        if(abs(x[k+1][i],y[k+1][i]) >epsilon) return;
        if((abs(y[k+1][i],y[k][i]))*p>epsilon) return;
    }
    problemSolved = true;
}
```

```
function updateY_float() public {
      // compute y[k+1][i], l[k+1][i] if p int

    for(uint i=0; i<N; i++){
        if(i==0) z[k][i]=x[k+1][i]+(l[k][i])*p;
        else z[k][i]=z[k][i-1]+x[k+1][i]+(l[k][i])*p;

    }

     h[k]=(((Pd)-z[k][N-1]))/(p*int(N));

    for(uint i; i<N; i++){

      y[k+1][i]=(h[k]*p)+x[k+1][i]+(l[k][i])*p;
      l[k+1][i]=l[k][i]+(x[k+1][i]-y[k+1][i])/p;
    }


    //check for convergence

    for(uint i=0; i<N; i++){
        if(abs(x[k+1][i],y[k+1][i]) >epsilon) return;
        if(((abs(y[k+1][i],y[k][i]))/p)>epsilon) return;
    }
    problemSolved = true;

}


function abs(int a, int b) public pure returns (int){
    if(a>=b) return a-b;
    return b-a;
}

function stillWaiting () view  public returns (bool) {
    for (uint8 i=0; i<whitelist.length; i++){
        if ( waiting[ whitelist[i] ] ){ return true; }
    }
    return false;
    }

function resetWaiting () public returns(bool){
// Reset the boolean waiting for each address
    for(uint8 i=0; i<whitelist.length; i++){
        waiting[whitelist[i]] = true;
    }
    return true;
    }
```

```
    function reset() public {
    // Helper function in order to solve multiple times.
        k = 0;
        problemSolved = false;
        resetWaiting();
        init=true;
            return;
        }


}
```