Ένας νέος αλγόριθμος συμπίεσης δεδομένων

Κωνσταντινίδης Νεκτάριος, Ευαγγελίδης Γεώργιος, Σαμαράς Νικόλαος

Τμήμα Εφαρμοσμένης Πληροφορικής, Πανεπιστήμιο Μακεδονίας, Εγνατία 156
{it0316, gevan, samaras}@uom.gr

Abstract

During the recent years there has been a revolution in computer science world-wide. New technologies have offered an increase in data transfer speeds and in data storage space. This increase however can not satisfy the demand for speed and storage space which increases more rapidly. For this reason data compression in data manipulation is very important and can be applied to any situation where data is used. With data compression storage of data requires less space and can be transferred faster. In this paper a new compression algorithm is presented. We explain how it works and we compare it to other well-known algorithms. Although this new compression algorithm is at an early stage we believe it has great potential.

Keywords: Algorithm, Data, Compression.

1. Εισαγωγή

Την τελευταία δεκαετία πραγματοποιείται ένα άλμα τεχνολογικής εξέλιξης στον τομέα της πληροφορικής. Η χωρητικότητα αποθήκευσης και η ταχύτητα μεταφοράς δεδομένων αυξάνεται σταθερά με τις νέες τεχνολογίες. Παρόλα αυτά, οι απαιτήσεις σε χωρητικότητα και ταχύτητα αυξάνονται με μεγαλύτερο ρυθμό απ' ότι η εξέλιξή τους. Γι' αυτόν τον λόγο η συμπίεση δεδομένων είναι σημαντικός τομέας στην διαχείριση πληροφοριών και εφαρμόζεται παντού. [Khalid Sayood(2000)]

Η συμπίεση μειώνει το μέγεθος ενός αρχείου. Αυτό βοηθάει στην αποθήκευση δεδομένων και στην γρήγορη μεταφορά τους. Η χωρητικότητα αποθήκευσης δεδομένων κοστίζει, οπότε ένα συμπιεσμένο αρχείο είναι πιο «φθηνό» από ένα ασυμπίεστο. Επίσης, στην μεταφορά δεδομένων τα μικρότερα αρχεία μεταφέρονται ταχύτερα απ' ότι τα μεγαλύτερα. Οπότε ένα συμπιεσμένο αρχείο μεταφέρεται πιο γρήγορα απ' ότι ένα ασυμπίεστο. Η συμπίεση λοιπόν έχει ως αποτέλεσμα την εξοικονόμηση χρόνου, υπολογιστικών πόρων και κατά συνέπεια χρημάτων.

Υπάρχουν δύο είδη συμπίεσης, η απωλεστική και η μη απωλεστική. Στην απωλεστική συμπίεση ένα συμπιεσμένο αρχείο δεν μπορεί να αποσυμπιεστεί στην αρχική μορφή του, λόγω του ότι έχουν χαθεί κάποια δεδομένα. Για αυτόν τον λόγο η μέθοδος αυτή δεν χρησιμοποιείται σε σημαντικά δεδομένα όπου πρέπει να αποσυμπιεστούν στην αρχική τους μορφή. Χρησιμοποιείται κυρίως για «Digitally Sampled Analog Data» (DSAD), δηλαδή δεδομένα ήχου, video, γραφικών ή εικόνων.

Υπάρχουν διάφοροι αλγόριθμοι, αλλά οι πιο πολλοί χρησιμοποιούν την αποκοπή δεδομένων εκτός ορίων. Αυτό σημαίνει πως επιλέγονται κάποια όρια και τα δεδομένα που βρίσκονται εκτός αυτών αποκόπτονται. Για παράδειγμα ένα αρχείο ήχου μπορεί να συμπιεστεί αφαιρώντας τους ήχους που δεν μπορεί να ακούσει ο άνθρωπος, δηλαδή τους ήχους με πολύ χαμηλές ή πολύ υψηλές συχνότητες. Γνωστοί απωλεστικοί αλγόριθμοι συμπίεσης είναι οι JPEG, MPEG και INDEO. [Ricardo Baeza-Yates, Berthier Ribeiro-Neto (1999)]

Στη μη απωλεστική συμπίεση το συμπιεσμένο αρχείο όταν αποσυμπιεστεί είναι ακριβώς το ίδιο με το αρχικό. Η μη απωλεστική συμπίεση ψάχνει για επαναλαμβανόμενες δυαδικές δομές τις οποίες κατόπιν κωδικοποιεί με μικρότερο μέγεθος. Λόγω του ότι εξαρτάται από τον αριθμό των επαναλαμβανόμενων δομών δεν λειτουργεί καλά σε τυχαία δεδομένα. Εφαρμόζεται κυρίως σε κείμενο. Οι περισσότεροι αλγόριθμοι για μη απωλεστική συμπίεση βασίζονται στην μέθοδο συμπίεσης LZ που δημιουργήθηκε από τον Lempel και τον Ziv. [Ziv and Lempel, (1978), Ziv, (1978), Welch, (1984)]

Στην παρούσα εργασία παρουσιάζεται ένας νέος αλγόριθμος (Nek-Kon) για τη συμπίεση και την αποσυμπίεση δεδομένων που αποτελείται από δύο φάσεις, τη φάση Nek και τη φάση Kon. Ο αλγόριθμος είναι μη απωλεστικός και μπορεί να εφαρμοστεί σε όλα τα είδη δυαδικών δεδομένων (κείμενο, ήχο, εικόνα, video κτλ.).

Η περιγραφή της λειτουργίας του αλγόριθμου γίνεται στις ενότητες 2 και 3 όπου αναλύεται η διαδικασία της συμπίεσης και της αποσυμπίεσης αντίστοιχα. Στην ενότητα 4 περιγράφονται τα αποτελέσματα της συμπίεσης αρχείων διαφόρων ειδών και μεγεθών με την χρήση του αλγόριθμου και γίνεται σύγκριση με άλλα εμπορικά προγράμματα συμπίεσης. Στην ενότητα 5 παρουσιάζουμε τα συμπεράσματά μας και σκέψεις για μελλοντική έρευνα.

Λόγω χώρου δεν είναι δυνατό να αναλυθεί πλήρως η διαδικασία της συμπίεσης και της αποσυμπίεσης. Στην διεύθυνση http://algor.uom.gr/texts/bsc/Nek-Kon.pdf υπάρχει ολόκληρη η εργασία όπου αναλύεται πλήρως μέσα από ένα παράδειγμα.

2. Συμπίεση

Η φάση Nek εφαρμόζεται πρώτη στα δεδομένα και μετατρέπει την μορφή των δεδομένων σε μια συμπυκνωμένη μορφή δημιουργώντας επαναλαμβανόμενες δομές και συμπιέζοντας ταυτόχρονα. Η φάση Kon εφαρμόζεται στη συνέχεια και αρχικά εντοπίζει και επιλέγει τις δομές που παρέχουν την καλύτερη συμπίεση. Έπειτα συμπιέζει τα δεδομένα στην τελική τους μορφή με βάση τις δομές που επιλέχθηκαν. Το τελικό αρχείο που προκύπτει ύστερα από την εφαρμογή των δύο φάσεων είναι το ιστορικό. Η αποσυμπίεση των δεδομένων εφαρμόζεται στο ιστορικό με μια σάρωση (Ενότητα 3).

2.1 Φάση Nek

Η φάση Nek ξεκινά σαρώνοντας τα δεδομένα. Κατά την σάρωση αποθηκεύονται σε έναν πίνακα, μικρών διαστάσεων, πληροφορίες που θα χρειαστούν για την μετατροπή των δεδομένων σε μια νέα συμπιεσμένη μορφή με περισσότερα μοτίβα (patterns).

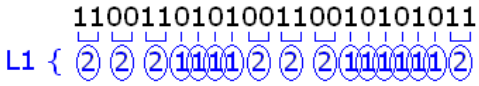
Ο πίνακας που δημιουργείται κατά την σάρωση επεξεργάζεται ώστε να αποφασιστεί πώς θα μεταβληθεί η αρχική μορφή των δεδομένων. Ο σκοπός των μεταβολών είναι η δημιουργία μοτίβων και η συμπίεση των αρχικών δεδομένων. Ύστερα από την επιλογή και την εφαρμογή των μεταβολών θα χρειαστεί να αποθηκευτούν ορισμένα δεδομένα τα οποία είναι απαραίτητα για την αποσυμπίεση. Αυτά αποθηκεύονται ξεχωριστά σε ένα ιστορικό. Τα στάδια που ακολουθεί ο αλγόριθμος είναι τα εξής:

2.1.1 Σάρωση των δεδομένων και δημιουργία του πίνακα.
2.1.2 Επεξεργασία του πίνακα
2.1.3 Επιλογή των μεταβολών που θα εφαρμοστούν στα δεδομένα.
2.1.4 Αποθήκευση των απαραίτητων πληροφοριών που απαιτούνται για την αποσυμπίεση.
2.1.5 Μετατροπή των δεδομένων στην νέα τους μορφή.

2.1.1 Σάρωση των δεδομένων και δημιουργία του πίνακα

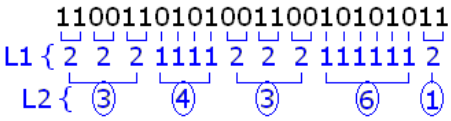
Κατά την σάρωση των δεδομένων δημιουργείται ένας πίνακας ο οποίος περιέχει τις τιμές των μεταβλητών L1, L2 και N. Ο πίνακας είναι ταξινομημένος πρώτα ως προς L1 και ύστερα ως προς L2 σε αύξουσα σειρά.

Η μεταβλητή L1 αντιπροσωπεύει το μήκος των συνεχόμενων bit με την ίδια τιμή. Στα παρακάτω δεδομένα (Εικόνα 1) το πρώτο L1 έχει τιμή δύο, αφού τα bit ξεκινάνε με δύο συνεχόμενους άσσους. Το δεύτερο L1 έχει τιμή πάλι δύο αφού ακολουθούν, μετά το προηγούμενο L1, δύο συνεχόμενα μηδενικά. Η διαδικασία αυτή συνεχίζεται ως το τέλος των δεδομένων βρίσκοντας έτσι τα L1.



Εικόνα 1. *Τα L1 των δεδομένων*

Η μεταβλητή L2 αντιπροσωπεύει το μήκος των συνεχόμενων ίδιων L1. Στα δεδομένα του παραδείγματος το πρώτο L2 έχει τιμή τρία, αφού τα bit ξεκινάνε με τρία συνεχόμενα L1 με τιμή δύο. Το δεύτερο L2 έχει τιμή τέσσερα αφού ακολουθούν, μετά το προηγούμενο L2, τέσσερα συνεχόμενα L1 με τιμή ένα. Η διαδικασία αυτή συνεχίζεται ως το τέλος των δεδομένων βρίσκοντας έτσι τους συνδυασμούς L1-L2 (Εικόνα 2).



Εικόνα 2. *Τα L1-L2 των δεδομένων*

Η μεταβλητή N είναι το πλήθος των συνδυασμών L1-L2 που υπάρχουν στα δεδομένα (Πίνακας 1).

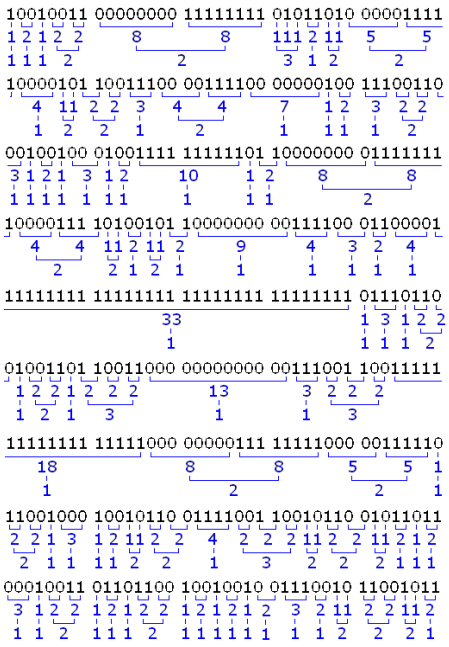
Πίνακας 1. *Ο πίνακας L1-L2 των δεδομένων*

i	L1	L2	N
1	1	4	1
2	1	6	1
3	2	1	1
4	2	3	2

Στον Πίνακα 2 και την Εικόνα 3 φαίνεται το αποτέλεσμα της σάρωσης ενός αρχείου 360 bit σύμφωνα με το προηγούμενο παράδειγμα. Στα παρακάτω θα χρησιμοποιούμε ως παράδειγμα τα δεδομένα της Εικόνας 3.

Πίνακας 2. *Ο πίνακας των L1-L2 του παραδείγματος*

i	L1	L2	N
1	1	1	21
2	1	2	9
3	1	3	1
4	2	1	18
5	2	2	11
6	2	3	3
7	3	1	10
8	4	1	4
9	4	2	2
10	5	2	2
11	7	1	1
12	8	2	3
13	9	1	1
14	10	1	1
15	13	1	1
16	18	1	1
17	33	1	1



Εικόνα 3. *Τα L1-L2 του παραδείγματος*

2.1.2 Επεξεργασία του πίνακα

Η επεξεργασία του πίνακα γίνεται με την προσθήκη στηλών. Οι στήλες που προστίθενται είναι οι στήλες FL1, FL2, sortN και Temp. Οι στήλες αυτές είναι απαραίτητες για την πραγματοποίηση των επόμενων βημάτων του αλγόριθμου.

Η στήλη FL1 περιέχει έναν μετασχηματισμό των τιμών της στήλης L1 τέτοιο ώστε οι νέες τιμές κάθε ομάδας ίδιων L1 να είναι μια ακολουθία συνεχόμενων ακέραιων αριθμών που ξεκινά από την τιμή 1.

Η στήλη FL2 περιέχει έναν μετασχηματισμό των τιμών της στήλης L2 τέτοιο ώστε οι νέες τιμές που αντιστοιχούν σε μια ομάδα ίδιων L1 να είναι μια ακολουθία συνεχόμενων ακέραιων αριθμών που ξεκινά από την τιμή 1.

Η στήλη sortN περιέχει τους δείκτες i, για κάθε συνδυασμό L1-L2, ταξινομημένους ως προς το N σε φθίνουσα σειρά. Σε περίπτωση που τα N είναι ίσα, τότε ταξινομούνται ως προς L1 κατά αύξουσα σειρά. Σε περίπτωση που τα N και τα L1 είναι ίσα, τότε ταξινομούνται ως προς L2 κατά αύξουσα σειρά.

Η στήλη Temp περιέχει τιμές που αποθηκεύονται προσωρινά κατά την εφαρμογή μεταβολών στα δεδομένα.

Ύστερα από την επεξεργασία του Πίνακα 2, προκύπτει ο παρακάτω Πίνακας 3.

Πίνακας 3. *Ο πίνακας του παραδείγματος ύστερα από επεξεργασία*

i	L1	L2	N	FL1	FL2	sortN	Temp
1	1	1	21	1	1	1	0
2	1	2	9	1	2	4	0
3	1	3	1	1	3	5	0
4	2	1	18	2	1	7	0
5	2	2	11	2	2	2	0
6	2	3	3	2	3	8	0
7	3	1	10	3	1	6	0
8	4	1	4	4	1	12	0
9	4	2	2	4	2	9	0
10	5	2	2	5	1	10	0
11	7	1	1	6	1	3	0
12	8	2	3	7	1	11	0
13	9	1	1	8	1	13	0
14	10	1	1	9	1	14	0
15	13	1	1	10	1	15	0
16	18	1	1	11	1	16	0
17	33	1	1	12	1	17	0

2.1.3 Επιλογή των μεταβολών που θα εφαρμοστούν στα δεδομένα.

Ύστερα από την σάρωση των δεδομένων, την δημιουργία και την επεξεργασία του πίνακα, επιλέγονται οι μεταβολές που θα υποστούν τα δεδομένα. Οι μεταβολές έχουν ως σκοπό να συμπιέσουν τα δεδομένα και να δημιουργήσουν περισσότερα μοτίβα μέσα σε αυτά.

Η πρώτη μεταβολή είναι η αντικατάσταση των συνδυασμών L1-L2 με τους συνδυασμούς FL1-FL2. Αυτό επιτυγχάνει πολύ καλή συμπίεση σε δεδομένα όπου τα L1 ή τα L2 δεν είναι συνεχόμενα και εφαρμόζεται πάντα. Με την μεταβολή αυτή οι συνδυασμοί συμπυκνώνονται συμπιέζοντας τα δεδομένα. Στο παράδειγμα τα αρχικά δεδομένα είχαν μέγεθος 360 bit. Ύστερα από την εφαρμογή αυτής της μεταβολής τα δεδομένα θα έχουν μήκος 289 bit, όπως φαίνεται παρακάτω.

```
10010011 00000000 11111111 01011010 00001111
10000101 10011100 00111100 00000100 11100110
00100100 01001111 11111101 10000000 01111111
10000111 10100101 10000000 00111100 01100001
11111111 11111111 11111111 11111111 01110110
01001101 10011000 00000000 00111001 10011111
11111111 11111000 00000111 11111000 00111110
11001000 10010110 01111001 10010110 01011011
00010011 01101100 10010010 01110010 11001011
```

Εικόνα 4. *Αρχικά δεδομένα*

```
10010011 00000001 01001011 11100001 01100111
00001111 00000010 01110011 00010010 00100111
11111101 10000000 11110000 10110100 11111111
00001110 01111000 00000000 01000100 11011001
00110011 11111111 00011001 10000000 00001111
11100000 10011011 10110100 11000011 00110100
11010010 01110110 01001001 10110110 11000110
10011010 0
```

Εικόνα 5. *Δεδομένα μετά την 1η μεταβολή*

Η δεύτερη μεταβολή εφαρμόζεται στους συνδυασμούς FL1-FL2 που δεν εμφανίζονται πολλές φορές στα δεδομένα. Τους αντικαθιστά όλους στον πίνακα, με έναν κοινό συνδυασμό FL1-FL2. Ο κοινός συνδυασμός FL1-FL2 είναι μικρού μήκους οπότε με την μεταβολή αυτή συμπιέζονται τα δεδομένα. Επίσης, τα δεδομένα μετά την $2^{η}$ μεταβολή περιέχουν μόνο συχνές δομές, δηλαδή στην ουσία περισσότερα μοτίβα από πριν. Λόγω χώρου, πλήρης ανάλυση της δεύτερης μεταβολής υπάρχει στην εργασία http://algor.uom.gr/texts/bsc/Nek-Kon.pdf.

2.1.4. Αποθήκευση των απαραίτητων πληροφοριών που απαιτούνται για την αποσυμπίεση

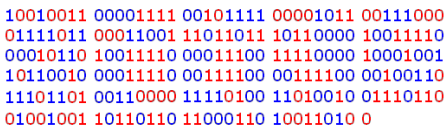
Οι απαραίτητες πληροφορίες που απαιτούνται για την αποσυμπίεση των δεδομένων στην αρχική τους μορφή είναι τα μήκη L1 και L2. Αν έχει εφαρμοστεί η $2^{η}$ μεταβολή τότε απαιτείται και η αποθήκευση των πρόσθετων πληροφοριών της $2^{ης}$ μεταβολής (2.1.3). Οι πληροφορίες αυτές αποθηκεύονται ξεχωριστά σε ένα ιστορικό.

2.1.5. Μετατροπή των δεδομένων στην νέα τους μορφή

Η μετατροπή των δεδομένων γίνεται σύμφωνα με τον βασικό πίνακα. Κατά την σάρωση των δεδομένων κάθε συνδυασμός L1-L2 αντικαθιστάται από τον αντίστοιχο συνδυασμό FL1-FL2. Αν ο συνδυασμός FL1-FL2 είναι ο κοινός συνδυασμός FL1-FL2 της $2^{ης}$ μεταβολής (2.1.3) τότε αποθηκεύονται στο ιστορικό πρόσθετες

πληροφορίες. Οι πρόσθετες πληροφορίες χρησιμοποιούνται κατά την αποσυμπίεση ώστε να είναι γνωστό κάθε κοινός συνδυασμός ποιον συνδυασμό FL1-FL2 είχε αντικαταστήσει. Ενδεικτικά, ακολουθούν τα δεδομένα μετά την $2^{η}$ μεταβολή.

Εικόνα 6. *Αρχικά δεδομένα*



Εικόνα 7. *Δεδομένα μετά την $2^{η}$ μεταβολή*

2.2 Φάση Kon

Ύστερα από την εκτέλεση της φάσης Nek παραμένουν το ιστορικό και τα δεδομένα που συμπιέστηκαν.

Η φάση Kon ξεκινά σαρώνοντας τα δεδομένα δημιουργώντας έναν πίνακα μοτίβων, όπου αποθηκεύονται τα πιο συχνά μοτίβα που περιέχουν τα δεδομένα. Ύστερα από την σάρωση δημιουργείται ένας νέος πίνακας, ο πίνακας ταξινόμησης. Ο πίνακας ταξινόμησης περιέχει μοτίβα που παρέχουν την μεγαλύτερη συμπίεση, ταξινομημένα ως προς αυτή κατά φθίνουσα σειρά. Κατά την επιλογή ενός μοτίβου υπολογίζονται και οι τυχόν συγκρούσεις ή περιπτώσεις που μοτίβα περιέχουν άλλα μοτίβα. Τα δεδομένα έπειτα ξανασαρώνονται. Όταν εντοπίζεται στα δεδομένα μοτίβο που υπάρχει στον πίνακα ταξινόμησης, τότε αντικαθίσταται με μια κοινή δομή αντικατάστασης και παράλληλα αποθηκεύεται στο ιστορικό πρόσθετες πληροφορίες. Το τελικό αρχείο που παραμένει ύστερα από την εφαρμογή των δύο φάσεων είναι το ιστορικό. Η αποσυμπίεση των δεδομένων εφαρμόζεται στο ιστορικό με μια σάρωση. Τα στάδια που ακολουθεί η φάση Kon είναι τα εξής:

2.2.1 Σάρωση των δεδομένων και δημιουργία πίνακα των πιο συχνών μοτίβων.
2.2.2 Ταξινόμηση των μοτίβων με βάση την συμπίεση που παρέχουν. Έλεγχος για σύγκρουση των μοτίβων και δημιουργία του πίνακα ταξινόμησης.
2.2.3 Σάρωση των δεδομένων αντικαθιστώντας τα μοτίβα
2.2.4 Αποθήκευση του τελικού ιστορικού.

2.2.1 Σάρωση των δεδομένων και δημιουργία του πίνακα των πιο συχνών μοτίβων.

Η φάση Kon κατά την εκτέλεση των βημάτων του χρησιμοποιεί κάποιους πίνακες και μεταβλητές. Ένας από τους πίνακες αυτούς είναι και ο πίνακας των μοτίβων που περιέχει τα συχνά μοτίβα και τον αριθμό των φορών που εντοπίστηκαν στα δεδομένα. Η φάση Kon ξεκινά σαρώνοντας τα δεδομένα και δημιουργώντας παράλληλα τον

πίνακα των μοτίβων. Τα μοτίβα είναι συνδυασμοί L1. Για παράδειγμα, το μοτίβο 001110 είναι το μοτίβο 2-3-1.

2.2.2 Ταξινόμηση των μοτίβων με βάση την συμπίεση που παρέχουν. Έλεγχος για σύγκρουση των μοτίβων και δημιουργία του πίνακα ταξινόμησης.

Ύστερα από την σάρωση των δεδομένων και την δημιουργία του πίνακα μοτίβων, τα μοτίβα ταξινομούνται ως προς την συμπίεση που παρέχουν. Δημιουργείται ένας νέος πίνακας, ο πίνακας ταξινόμησης, που περιέχει τα καλύτερα μοτίβα. Ο πίνακας ταξινόμησης δημιουργείται ακολουθώντας τρία βήματα. Πρώτα εξετάζεται αν η επιλογή κάθε μοτίβου αυξάνει ή μειώνει το τελικό μέγεθος των δεδομένων. Έπειτα από τα μοτίβα που μειώνουν το μέγεθος των δεδομένων επιλέγεται το καλύτερο. Τέλος, ελέγχεται αν υπάρχει σύγκρουση μεταξύ του μοτίβου που επιλέχθηκε και των μοτίβων του πίνακα μοτίβων. Τα μοτίβα, στον πίνακα των μοτίβων, στα οποία εντοπίζεται σύγκρουση μειώνεται ο αριθμός των φορών που εντοπίστηκαν. Επαναλαμβάνεται η διαδικασία από την αρχή ώσπου να μην βρεθεί μοτίβο που να μειώνει το μέγεθος των αρχικών δεδομένων.

Το πρόβλημα της σύγκρουσης

Λόγω του ότι το μήκος των μοτίβων είναι μεταβλητό και ο πίνακας περιέχει όλα τα συχνά μοτίβα, μπορεί αρκετά μοτίβα να «συγκρούονται». Για παράδειγμα από τα μοτίβα:

2-3-1 και 4-2-3-1-5 και 4-1-2-3

Το μοτίβο 4-2-3-1-5 «συγκρούεται» με το μοτίβο 2-3-1, και το μοτίβο 2-3-1 με το μοτίβο 4-1-2-3 μπορεί να «συγκρούονται» σε ορισμένες περιπτώσεις όπως αυτή:

…-4-1-2-3-1-…

Στα μοτίβα που «συγκρούονται» με το μοτίβο που έχει επιλεγεί πρέπει να μειώνεται ο αριθμός των φορών που έχουν εμφανιστεί ώστε να μην υπολογίζεται το ίδιο μοτίβο 2 φορές. Το πρόβλημα αυτό ονομάζεται πρόβλημα «σύγκρουσης».

Για την αντιμετώπιση του προβλήματος της σύγκρουσης αφού επιλεγεί ένα μοτίβο για τον πίνακα ταξινόμησης, στα μοτίβα του πίνακα μοτίβων που συγκρούονται με αυτό μειώνεται ο αριθμός που εμφανίστηκαν.

2.2.3 Σάρωση των δεδομένων αντικαθιστώντας τα μοτίβα

Στο βήμα αυτό τα μοτίβα αντικαθιστούνται από μια μικρότερη δομή αντικατάστασης.

Κατά την σάρωση των δεδομένων κάθε φορά που εντοπίζεται ένα μοτίβο αντικαθιστάται από την δομή αντικατάστασης ενώ παράλληλα αποθηκεύεται στο ιστορικό πρόσθετες πληροφορίες. Οι πρόσθετες πληροφορίες χρησιμοποιούνται κατά

την αποσυμπίεση ώστε να είναι γνωστό κάθε δομή αντικατάστασης ποιο μοτίβο είχε αντικαταστήσει.

Ακολουθούν τα δεδομένα πριν και μετά την εφαρμογή της φάσης Kon στο παράδειγμα. Παρατηρούμε ότι τα δεδομένα μειώθηκαν κατά 30bit. Λόγω χώρου, πλήρης ανάλυση της φάσης Kon υπάρχει στην εργασία http://algor.uom.gr/texts/bsc/Nek-Kon.pdf.

```
10010011 00001111 00101111 00001011 00111000
01111011 00011001 11011011 10110000 10011110
00010110 10011110 00011100 11110000 10001001
10110010 00011110 00111100 00111100 00100110
11101101 00110000 11110100 11010010 01110110
01001001 10110110 11000110 10011010 0
```

Εικόνα 7. *Πριν την φάση Kon*

```
10010011 00000110 10000010 11001110 00001001
11001100 01001000 10011110 11000001 01101001
11110001 10000010 00100110 11001000 00111000
00111110 11001000 10010110 01111101 00110100
10011101 10010010 01101101 10110001 10100110
100
```

Εικόνα 8. *Μετά την φάση Kon*

2.2.4 Αποθήκευση του τελικού ιστορικού.

Στο βήμα αυτό αποθηκεύεται το τελικό ιστορικό. Το τελικό ιστορικό περιλαμβάνει το ιστορικό της φάσης Nek, το ιστορικό της φάσης Kon και τα συμπιεσμένα δεδομένα.

Το ιστορικό της φάσης Kon περιλαμβάνει την αποθήκευση των μοτίβων και τις πρόσθετες πληροφορίες για να τα ξεχωρίζει (2.2.3).

3. Αποσυμπίεση

Η αποσυμπίεση των δεδομένων εφαρμόζεται στο ιστορικό με μια σάρωση. Κατά την διαδικασία της αποσυμπίεσης δημιουργούνται οι βασικοί πίνακες της συμπίεσης.

Διαβάζονται στην αρχή από το ιστορικό τα L1, L2 και οι πληροφορίες της $2^{ης}$ μεταβολής και δημιουργείται εύκολα ο βασικός πίνακας της φάσης Nek. Ύστερα διαβάζονται τα μοτίβα της φάσης Kon που είχαν αποθηκευτεί στο ιστορικό. Δημιουργείται ο πίνακας ταξινόμησης.

Τα δεδομένα σαρώνονται και γίνεται αντικατάσταση των δομών με βάση τους πίνακες που έχουν δημιουργηθεί. Πρώτα αντικαθιστάται με βάση το ιστορικό τα μοτίβα και ύστερα αντικαθιστούνται τα FL1-FL2 του πίνακα Nek με τα αντίστοιχα L1-L2.

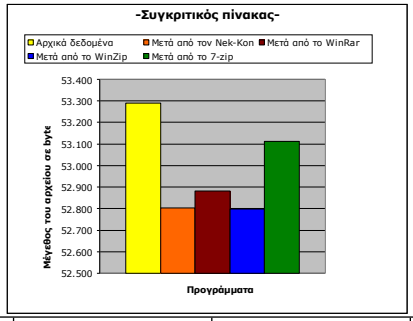
Δημιουργείται το αρχικό αρχείο.

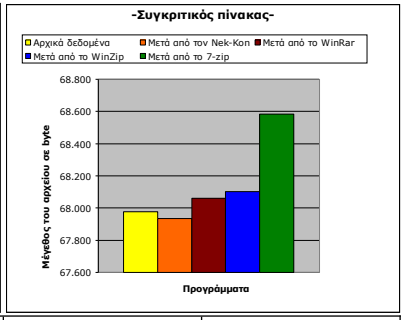
4. Εμπειρική Μελέτη

Στην ενότητα αυτή θα συγκριθεί το πρόγραμμα Nek-Kon με άλλα γνωστά εμπορικά προγράμματα συμπίεσης. Αν και το πρόγραμμα βρίσκεται σε πολύ αρχικό στάδιο, θα γίνει προσπάθεια για την ανάδειξη των δυνατοτήτων του.

Θα δοκιμαστούν αρχεία διαφόρων ειδών και μεγεθών. Για κάθε αρχείο θα γίνει ανάλυση των αποτελεσμάτων και των συμπερασμάτων που προκύπτουν. Τα εμπορικά προγράμματα που θα χρησιμοποιηθούν είναι το WinZip (έκδοσης 11.1), το WinRar (έκδοσης 3.62) και το 7-Zip (έκδοσης 4.57). Τα προγράμματα αυτά έχουν ρυθμιστεί με στόχο την καλύτερη συμπίεση των δεδομένων. Τα αρχεία που χρησιμοποιήθηκαν είναι αρχεία εικόνων, πολυμέσων και αρχεία συστήματος (*.dll). Τα αρχεία αυτά επιλέχτηκαν τυχαία από τον υπολογιστή και το διαδίκτυο. Για κάθε αρχείο παρουσιάζονται τα αποτελέσματα σε μορφή πίνακα, με γραφική σύγκριση και με σχολιασμό. Τα συμπιεσμένα αρχεία του Nek-Kon έχουν ελεγχτεί για την σωστή συμπίεσή τους, καθώς αποσυμπιέστηκαν και συγκρίθηκαν με τα αρχικά δεδομένα. Δεν παρατηρήθηκε απώλεια ή παραμόρφωση των αρχικών δεδομένων.

Γράφημα και πίνακας 1ου αρχείου ***Γράφημα και πίνακας 2ου αρχείου***





	Μέγεθος σε byte	Χρόνος σε sec	Μέγεθος σε byte	Χρόνος σε sec
Αρχικά δεδομένα	53.289		67.978	
Nek-Kon	52.804	208	67.936	614
WinRar	52.881	1	68.059	1
WinZip	52.798	1	68.104	1
7-zip	53.113	1	68.585	1

Γράφημα και πίνακας 3ου αρχείου ***Γράφημα και πίνακας 4ου αρχείου***

	Μέγεθος σε byte	Χρόνος σε sec	Μέγεθος σε byte	Χρόνος σε sec
Αρχικά δεδομένα	12.293.945		16.384	
Nek-Kon	12.241.359	46916	1.328	3
WinRar	12.205.543	6	1.052	1
WinZip	12.186.879	2	1.144	1
7-zip	12.256.518	5	1.062	1

Το πρώτο αρχείο είναι ο ήχος “rkflood01c.mp3” με μέγεθος 53.289 byte. Όπως παρατηρούμε από το γράφημα και από τα αποτελέσματα το αρχείο αυτό συμπιέστηκε καλύτερα από το πρόγραμμα WinZip με 52.798 byte .

Το δεύτερο αρχείο είναι η εικόνα “Lamborghini.jpg” με μέγεθος 67.978 byte. Όπως παρατηρούμε από το γράφημα και από τα αποτελέσματα, την καλύτερη συμπίεση έκανε το πρόγραμμα Nek-Kon με 67.936 byte. Παρατηρούμε ότι μόνο το πρόγραμμα Nek-Kon μείωσε το μέγεθος του αρχικού αρχείου καθώς τα υπόλοιπα προγράμματα το αύξησαν.

Το τρίτο αρχείο είναι το βίντεο “tigerwoods08.wmv” με μέγεθος 12.293.945 byte. Όπως παρατηρούμε από το γράφημα και από τα αποτελέσματα το αρχείο αυτό συμπιέστηκε καλύτερα από το πρόγραμμα WinZip με 12.186.879 byte.

Το τέταρτο αρχείο είναι το αρχείο συστήματος “Visual3D.EngineMedia.dll” με μέγεθος 16.384 byte. Όπως παρατηρούμε από το γράφημα και από τα αποτελέσματα το αρχείο αυτό συμπιέστηκε καλύτερα από το πρόγραμμα WinRar με 1.052 byte.

5. Συμπεράσματα

Ύστερα από την εμπειρική μελέτη του νέου αλγόριθμου Nek-Kon, παρατηρούμε ότι σε ορισμένα αρχεία είναι καλύτερος από τους υπάρχοντες αλγόριθμους και σε ορισμένα άλλα χειρότερος.

Ο αλγόριθμος βρίσκεται σε πολύ αρχική φάση. Οι βελτιώσεις που πρέπει να γίνουν στον αλγόριθμο έχουν καταγραφεί και πρόκειται να πραγματοποιηθούν σε μελλοντική εργασία. Μια από τις πρώτες βελτιώσεις που έχουν τεθεί ως στόχο είναι η μείωση του χρόνου που απαιτεί η φάση Kon. Η φάση Kon είναι η κύρια αιτία που καθυστερεί ο αλγόριθμος σε σύγκριση με τους υπόλοιπους αλγόριθμους. Εκτιμάται ότι μετά τις βελτιώσεις ο χρόνος που απαιτεί η φάση Kon θα μειωθεί και θα είναι στα ίδια επίπεδα με τους άλλους αλγόριθμους. Επίσης, κάποιες άλλες βελτιώσεις θα κάνουν τον αλγόριθμο να συμπιέζει τα αρχεία ακόμα περισσότερο.

Κατά την έρευνα που πραγματοποιήθηκε δεν βρέθηκε άλλος μη απωλεστικός αλγόριθμος που να αλλάζει την μορφή των δεδομένων σε πρώτη φάση, συμπιέζοντας ταυτόχρονα, για να συμπιέσει καλύτερα στην επόμενη. Λόγω αυτού του

χαρακτηριστικού του, ο αλγόριθμος Nek-Kon μπόρεσε να συμπιέσει αρχεία που δεν μπορούσαν να συμπιέσουν άλλοι μη απωλεστικοί αλγόριθμοι. Αυτό γιατί οι αλγόριθμοι αυτοί περιμένουν τα αρχεία να έχουν ήδη μοτίβα. Ο αλγόριθμος Nek-Kon δημιουργεί μοτίβα και ύστερα τα συμπιέζει. Αυτό φαίνεται στο δεύτερο πείραμα όπου μόνο ο αλγόριθμος Nek-Kon μείωσε το μέγεθος του αρχείου ενώ όλοι οι άλλοι αλγόριθμοι το αύξησαν.

6. Αναφορές

1. Salomon, F. (2004). *"Data Compression, The Complete Reference 3rd edition",* Springer-Verlag,
2. Ziv, J., Lempel, A. (1978). *"Compression of Individual Sequences Via Variable-Rate Coding"*. IEEE Transactions on Information Theory, Vol. IT-25, pp. 530–536.
3. Ziv, J. (1978). *"Coding theorems for individual sequences"*. IEEE Transactions on Information Theory, Vol. IT-24, pp. 405–412.
4. Welch, T. A. (1984). *"A technique for high-performance data compression"*. Computer, Vol. 17, pp. 8–19.
5. M. Nelson and J.-L. Gailly, (1995) *"The Data Compression Book",* M&T Books
6. Khalid Sayood, (2002) *"Lossless Compression Handbook (Communications, Networking and Multimedia)",* Academic Press
7. Khalid Sayood, (2000) *"Introduction to Data Compression",* Morgan Kaufmann
8. Ricardo Baeza-Yates, Berthier Ribeiro-Neto, (1999) *"Modern Information Retrieval"*, Addison Wesley