Οδηγίες Κώδικα

Συναρτήσεις

Task A

marginal: Υπολογίζει την marginal συνάρτηση – διάνυσμα ως προς μια μεταβλητή όρισμα της συνάρτησης με την μέθοδο των factor graphs για δέντρο. Δέχεται σαν ορίσματα τις γενικές πράξεις πρόσθεσης $\oplus$, πολλαπλασιασμού $\otimes$, τις υποσυναρτήσεις που παραγοντοποιούν την βασική συνάρτηση, τον bipartite γράφο που συνδέει τις υποσυναρτήσεις με τα ορίσματα τους, το όνομα της μεταβλητής υπό την οποία θα γίνει το marginalization καθώς και ένα διάνυσμα με το σύνολο τιμών της.

summation: Υλοποιεί τη γενική πράξη πρόσθεσης $\oplus$ και μπορεί να δεχθεί πολλαπλές scalar τιμές, διανύσματα ή πίνακες και να κάνει την γενική άθροιση προς ολα τα στοιχεία τους.

multiplication: Υλοποιεί τη γενική πράξη πολλαπλασιασμού $\otimes$. και μπορεί να δεχθεί πολλαπλές scalar τιμές, διανύσματα ή πίνακες και να κάνει τον γενικό πολλαπλασιασμό προς ολα τα στοιχεία τους.

Task B

decode: Διορθώνει την κωδικολέξη που στάλθηκε στον δέκτη μέσω ενός δυαδικού συμμετρικού καναλιού με σκοπό να ανακτήσει την σωστή κωδικολέξη που έστειλε ο πομπός. Για τον σκοπό αυτό υλοποιεί τον επαναληπτικό αλγόριθμο για bit error correction σε factor graph με κύκλους. Δέχεται σαν ορίσματα το κανάλι επικοινωνίας, τον parity check matrix του κώδικα, τα σύμβολα του δέκτη καθώς και την κωδικολέξη που έλαβε.

BlockMapDecoding: Χρησιμοποιήθηκε σαν συνάρτηση ελέγχου για να δούμε εάν η λειτουργία της decode είναι σωστή. Υλοποιεί το ML κριτήριο για block decoding και υπολογίζει με brute force την πιο πιθανή κωδικολέξη που στάλθηκε.

Task C

distributionPairLDPC: Παράγει τις βέλτιστες κατανομές λ, ρ με τις οποίες θα μπορέσουμε να πλησιάσουμε την θεωρητική χωρητικότητα του δυαδικού συμμετρικού καναλιου με διαγραφές. Επίσης σαν έξοδο μας δίνει και το θεωρητικό κατώφλι σφάλματος του καναλιού error BP καθώς και το delta gap. Δέχεται σαν ορίσματα τους μέγιστους βαθμούς συνδεσιμότητας των ακμών σε κόμβους μεταβλητών και συναρτήσεων καθώς την πιθανότητα σφάλματος bit του καναλιού.

histogramPairLDPC: Παράγει τα προσεγγιστικά βέλτιστα διακριτά ιστογράμματα Λ, Ρ μέσω των βέλτιστων κατανομών λ, ρ που υπολογίσαμε μέσω της distributionPairLDPC. Σάν

ορίσματα δέχεται τις βέλτιστες κατανομές λ, ρ καθώς και τον αριθμό των μεταβλητών που θέλουμε να έχουμε δηλαδή το μήκος των κωδικολέξεων που θέλουμε να στέλνουμε.

LDPC: Παράγει ένα σύνολο από LDPC parity check matrices μέσω των διακριτών ιστογραμμάτων Λ, Ρ που υπολογίσαμε μέσω της histogramPairLDPC. Δέχεται σαν ορίσματα τα ιστογράμματα Λ, Ρ καθώς και τον αριθμό των LDPC πινάκων που θέλουμε να παράξουμε.

errorBP: Υπολογίζει μέσω binary search το θεωρητικό φράγμα της πιθανότητας σφάλματος ενός δυαδικού συμμετρικού καναλιου με διαγραφές δεδομένων κατανομών λ, ρ.

smartRound: Υλοποίει την βέλτιστη στρογγυλοποίηση ενός διανύσματος διατηρώντας σταθερό το ακαριαίο άθροισμα των στοιχείων του και ελαχιστοποιώντας το σφάλμα στρογγυλοποίησης.

Task A

Λεπτομέρειες

marginal:
Τα ορίσματα της έχουν της εξής δομές και περιγραφές στην MATLAB

variable_vector: είναι ένα διάνυσμα του σύνολού τιμών της μεταβλητής ex [0 1]

variable_name: είναι ένα string με το όνομα της μεταβλητής ex "x_1"

Bibartite_Matrix: ένας πίνακας που περιέχει 0 ή 1 ανάλογα με το εάν μια συνάρτηση συνδέεται με μια μεταβλητή δηλαδή εάν την έχει σαν όρισμά της. Ο πίνακάς αυτός ταυτίζεται με τον parity check matrix και περιέχει ουσιαστικά την πληροφορία του bipartite γράφου.

factorization_functions: ένα cell που περιέχει τα function handles των υποσυναρτήσεων που παραγοντοποιούν την συνάρτηση που θέλουμε να κάνουμε marginalization.

general_summation: function handle της συνάρτησης γενικού αθροίσματος

general_multiplication: function handle της συνάρτησης γενικού πολλαπλασιασμού

Ο τρόπος υλοποίησης γίνεται με τα εξής βήματα:
1. Μετατρέπουμε την πληροφορία του bipartite_matrix σε δέντρο με κορυφή την μεταβλητή που θέλουμε

2. Ξεκινώντας από τον κόμβο μεταβλητή στην κορυφή του δέντρου πάμε αναδρομικά στα κατώτερα στρώματα που συνδέονται διαδοχικά μέχρι να φτάσουμε στα φύλλα

3. Όταν φθάσουμε στα φύλα και έχουμε συλλέξει όλη την πληροφορία αρχίζουμε και εφαρμόζουμε τους message passing κανόνες ανάλογα τον κόμβο (συνάρτηση ή μεταβλητή) από κάτω προς τα πάνω μέχρι να ανακατασκευάσουμε το μήνυμα-διάνυσμα της μεταβλητής στην κορυφή.

Παρατηρήσεις

Ακολουθήθηκαν vectorization τεχνικές για να πετύχουμε επιτάχυνση του κώδικα MATLAB πάνω στην υλοποίηση των κανόνων μεταφοράς μηνυμάτων – διανυσμάτων.

$$\mu_0(x_0) = \prod_{\kappa=1}^{K} \mu_\kappa(x_\kappa)$$

Αρχικά πρέπει να υπολογίσουμε τον παραπάνω κανόνα που στην θέση του πολλαπλασιασμού έχουμε τον γενικό πολλαπλασιασμού $\otimes$ και στην θέση των συναρτήσεων μνημάτων $\mu_\kappa(x_\kappa)$ έχουμε μηνύματα - διανύσματα $\underline{\mu_\kappa}$ αφού το $\underline{x_\kappa}$ το υλοποιούμε σαν διάνυσμα. Για τον σκοπό αυτό μετατρέπουμε την παραπάνω εξίσωση στην μορφή

$$\underline{\mu_0} = \underline{\mu_1} \otimes \underline{\mu_2} \otimes \dots \otimes \underline{\mu_K}$$

Οπού στην προκειμένη περίπτωση ο γενικός πολλαπλασιασμος $\otimes$ εφαρμόζεται στοιχείο προς στοιχείο στα μημύματα – διανύσματα.

$$\mu_0(\chi_0) = \sum_{\sim x_0} f(x_0, x_1, \dots, x_K) \prod_{\kappa=1}^{K} \mu_\kappa(x_\kappa)$$

Για την υλοποίηση του παραπάνω κανόνα όπου στην θέση του αθροίσματος και του πολλαπλασιασμού έχουμε τις γενικές πράξεις $\oplus$, $\otimes$ αντίστοιχα και στην θέση των μεταβλητών έχουμε διανύσματα $\underline{\chi_\kappa}$ κάναμε τις εξής διαδικασίες.

Αρχικά δημιουργήσαμε τον πολυδιάστατο πίνακα F , διάστασης K+1 που δίνεται από την παρακάτω μορφή και με $\underline{x_\kappa}(i_\kappa)$ συμβολίζουμε την τιμή του διανύσματος $\underline{x_\kappa}$ στην θέση που δείχνει ο δείκτης i_κ

$$F = \{ f\left(\underline{x_0}(i_0), \underline{x_1}(i_1), \dots, \underline{x_K}(i_K) \right) \mid \forall i_0, i_1, \dots, i_K \}$$

Στη συνέχεια δημιουργούμε τον πολυδιάστατο πίνακα M, διάστασης K που δίνεται από την παρακάτω μορφή και με $\underline{\mu_\kappa}(i_\kappa) = \mu_\kappa(\underline{x_\kappa}(i_\kappa))$ συμβολίζουμε την τιμή του μηνύματος διανύσματος $\underline{\mu_\kappa}(i_\kappa)$ στην θέση που δείχνει ο δείκτης i_κ

$$M = \{ \underline{\mu_1}(i_1) \otimes \underline{\mu_2}(i_2) \otimes \dots \otimes \underline{\mu_K}(i_K) \mid \forall i_0, i_1, \dots, i_K \}$$

Τελικά για να κάνουμε τις γενικές πράξεις του αθροίσματος και πολλαπλασιασμού έχουμε τους μετασχηματισμούς:

A. Μετατρέπουμε τον F σε πίνακα με την πρώτη διάσταση να αντιστοιχεί στα στοιχεία $\underline{x_0}$ πάνω στα οποία θέλουμε να εξάγουμε το μήνυμα διάνυσμα $\underline{\mu_0}$

B. Μετατρέπουμε τον M σε διάνυσμα

C. Υλοποιούμε την πράξη $\underline{\mu_0} = \{ \bigoplus (F(i_0, :) \otimes M^T) \mid \forall i_0 \}$ όπου ο γενικός πολλαπλασιασμός $\otimes$ γίνεται στοιχείο προς στοιχείο στα δύο διανύσματα γραμμής και το γενικό άθροισμα $\bigoplus$ γίνεται πάνω σε όλα τα στοιχεία του διανύσματος που θα προκύψει.

Task B

Λεπτομέρειες

Decode:
Τα ορίσματα της έχουν της εξής δομές και περιγραφές στην MATLAB

received_codeword: ένα string που περιλαμβάνει τα σύμβολα που έλαβε ο δέκτης ex «0101011"
reciver_symbols: το σύνολο των συμβόλων που μπορεί να δεχθεί ο δέκτης ως string

Channel_Matrix: πίνακας που περιλαμβάνει τις υπό συνθήκη πιθανότητες του καναλιού

Bipartite_Matrix: ένας πίνακας που περιέχει 0 ή 1 ανάλογα με το εάν μια συνάρτηση συνδέεται με μια μεταβλητή δηλαδή εάν την έχει σαν όρισμά της. Ο πίνακάς αυτός ταυτίζεται με τον parity check matrix και περιέχει ουσιαστικά την πληροφορία του bipartite γράφου.

channel_messages: διάνυσμα που περιλαμβάνει τα κατά παραγγελία log ratio μηνύματα των καναλιών και η default τιμή είναι κενή. Χρησιμοποιήθηκε για να πειραματιστούμε με τον κώδικα και να καταλάβουμε την λειτουργία του

Παρατηρήσεις

Θα κάνουμε μια σύντομη περιγραφή της λειτουργίας του κώδικα. Οι παρακάτω δομές είναι οι κυρίες δομές που θα μας βοηθήσουν να καταλάβουμε την διαδικασία

channel_messages: διάνυσμα που χρησιμοποιούμε για να αποθηκεύσουμε τα log ratio μνήματα που υπολογίζουμε από το channel matrix, η που τα εισάγουμε έτοιμα.

Variable_Messages: Πίνακας που χρησιμοποιούμε για να σώσουμε τα log ration μνήματα από έναν κόμβο μεταβλητής προς έναν οποιαδήποτε κόμβο συνάρτησης. Η πρώτη διάσταση αναφέρετε στις μεταβλητές και η δεύτερη στις συναρτήσεις. Για κόμβους μεταβλητης-συναρτησης που δεν συνδέονται απλά βάζουμε μηδέν και αγνοούμε τις τιμές τους

Function_Messages: Πίνακας που χρησιμοποιούμε για να σώσουμε τα log ration μνήματα από έναν κόμβο συνάρτησης προς έναν οποιαδήποτε κόμβο μεταβλητής. Είναι ουσιαστικά ο ανάστροφος του παραπάνω πίνακα, απλά τον χρειαζόμαστε σαν ξεχωριστό.

connected_fun_to_var: ένας λογικός πίνακας που περιλαμβάνει την πληροφορία για την συνδεσιμότητα των κόμβων συνάρτησης με τους κόμβους μεταβλητών. Μέσω αυτού μπορούμε να επιταχύνουμε την αναζήτηση και να αγνοήσουμε τους κόμβους πού δεν συνδέονται . Ταυτίζεται με τον Parity Check Matrix.

Αλγοριθμική Διαδικασία υλοποίησης της decode

1. Υπολογίζουμε τα log ratio μηνύματα από τους κόμβους καναλιού στους κόμβους μεταβλητών και τα αποθηκεύουμε στο channel_messages

2. Αρχικοποιούμε τον πίνακα Variable_Messages με τα μηνύματα από το channel_messages τοποθετώντας τα στις σωστές θέσεις.

3. Ξεκινάμε την επαναληπτική διαδικασία που αποτελείται από τα εξής κομμάτια

    a) Υπολογίζουμε τα log ration μηνύματα των συναρτήσεων προς τις συνδεδεμένες μεταβλητές και τα σώζουμε στον πίνακα Function_Messages

    b) Υπολογίζουμε το άθροισμα των log ratio μηνυμάτων που φθάνουν σε μια μεταβλητή από τις συναρτήσεις μέσω του Function_Messages καθώς την επίδραση του κανάλου μέσω του channel_messages. Αποφασίζουμε ανάλογα το αποτέλεσμα αν θα διορθώσουμε το bit ή όχι.

    c) Υπολογίζουμε τα log ration μηνύματα των μεταβλητών προς τις συνδεδεμένες συανρτησεις και τα σώζουμε στον πίνακα Variable_Messages

    d) Ελέγχω αν η νέα κωδικολέξη επαληθεύει τον Parity Check Matrix ή όχι ,για να σταματήσω την επανάληψη

Task C

Λεπτομέρειες

distributionPairLDPC:

Για τον υπολογισμό των βέλτιστων λ, ρ υλοποιήσαμε την εξής προσεγγιστική λύση

i. Επιλέγουμε αρχικές συνθήκες $\rho *$ σύμφωνα με την μορφή της τελευταίας διαφάνειας για διάφορες τιμές $r_{avg} = 1 : r_{step} : r_{max}$ καθώς βοηθάει σε πιο γρήγορη σύγκληση των τιμών. (εφόσον υποθετικά αυτή η μορφή των $\rho *$ μπορεί να είναι και βέλτιστη)

ii. διακριτοποιούμε το x στο διάστημα $x = 1 : x_{step} : channel\_error$ καθώς για x πάνω απο το channel error η συνθήκη ικανοποιείται. Έτσι το πρόβλημα μετατρέπεται σε γραμμικό και μπορεί να λυθεί με την συνάρτηση της matlab "**linprog**"

iii. τρέχουμε τον αλγόριθμο βελτιστοποίησης ως προς λ για κάθε δεδομένη τιμή $ravg$ και σώζουμε τον ρυθμό που πέτυχε το ζευγάρι (λ, ρ)

iv. Επιλέγουμε ως βέλτιστο ζευγάρι (λ, ρ) το ζευγάρι που πέτυχε τον μέγιστο ρυθμό

histogramPairLDPC:

Αφού υπολογίσουμε με την παραπάνω διαδικασία τα βέλτιστα λ, ρ πρέπει στην συνέχεια να παράξουμε τα ιστογράμματα Λ, Ρ υπό τους εξής περιορισμούς:

i. Τα Λ,Ρ είναι ακέραια ιστογράμματα

ii. Πρέπει παράγουν τον ίδιο αριθμό ακμών

iii. Πρέπει η Λ να έχει άθροισμα ίσο με n, οπου n ο αριθμός των μεταβλητών, που το επιλέγουμε εμείς

Οπότε το πρόβλημα είναι το εξής, δεδομένου κάποιων λ,ρ πως θα παράξουμε τα Λ,Ρ με το ελάχιστο τετραγωνικό σφάλμα διακριτοποίησης δεδομένου των παραπάνων περιορισμών. Στο πρόβλημα αυτό βοήθησε η συνάρτηση της MATLAB intlinprog καθώς και η τεχνική The Cutting-Plane Method for Solving Convex Programs από το άρθρο mixed Integer Quadratic Programming Portfolio Optimization: Problem-Base για να μετατρέψουμε το πρόβλημα από Quadratic σε γραμμικό προγραμματισμό.

error_BP:

Για τον υπολογισμό του άνω φράγματος της πιθανότητας σφάλματος του δυαδικού συμμετρικού καναλιού με διαγραφή έχουμε τα εξής

1. Για $e < errorBP$ ισχύει ότι $\forall x \in [0,1]$ η συνθήκη $f(x,e) - x \leq 0$

2. Για $e > errorBP$ ισχύει ότι υπάρχουν $x \in (0,1]$ (εκτός του $x = 0$) για τα οποία η συνθήκη $f(x,e) - x > 0$

3. Για $e = errorBP$ υπάρχει μοναδικό $x \in (0,1]$ για το οποίο η συνθήκη $f(x,e) - x = 0$

4. Από τα παραπάνω συν το γεγονός ότι η συνάρτηση $f(x,e) - x$ είναι αύξουσα συνάρτηση του e μπορούμε με ένα binary search να βρούμε το $errorBP$

Παρατηρήσεις

Διαπιστώθηκε ότι για διάφορες αρχικές συνθήκες ρ* ο αλγόριθμος **δεν** συγκλίνει και πετάει σφάλμα οπότε με τις εντολές try/catch μπορούμε να καταστείλουμε τα σφάλματα. Επίσης όσο μεγαλύτερες τιμές για channel error και rmax έχουμε τόσο πιο δύσκολο ήταν να συγκλίνει ο αλγόριθμος

Λόγο της διακριτοποίησης του x υπάρχουν θέματα κοντά στο μηδέν και το βιβλίο προτείνει την εισαγωγή μιας συνθήκης σταθεροποίησης για το λ στο "Modern Coding Theory" σελ 114