Δάσκος Ραφαήλ – Α.Μ.: 03116049
Αλγόριθμοι και Πολυπλοκότητα
Ροή Λ: Λογισμικό Η/Υ
Σ.Η.Μ.Μ.Υ. – Ε.Μ.Π. – $7^{ο}$ εξάμηνο
$4^{η}$ Σειρά Γραπτών Ασκήσεων

Άσκηση 1: Επιβεβαίωση και Αναπαράσταση Συντομότερων Μονοπατιών

(α) Έστω ότι έχουμε όλα τα $δ_i$ από τον κόμβο 1 προς τους υπόλοιπους του γράφου μας. Για να δούμε αν όντως είναι σωστές αυτές οι αποστάσεις θα πρέπει να εφαρμόσουμε μια παραλλαγή του αλγορίθμου BFS. Θα ξεκινήσουμε από την κορυφή 1 και θα κοιτάξουμε όλες τις γειτονικές κορυφές. Αν μια από αυτές έχει βάρος μικρότερο από το $δ_i$, τότε δεν είναι σωστές οι αποστάσεις που μας έχουν δοθεί. Αν είναι μεγαλύτερο το βάρος από το $δ_ι$ θα περιμένουμε καθώς μπορεί κατά τη διάσχιση να παίρνει και κάποια ακμή με αρνητικό μήκος και άρα το τελικό ελάχιστο μονοπάτι να είναι μικρότερο. Αν είναι ίσα τότε παίρνω τον κόμβο που είχε την ακμή στο τελικό γράφημα. Ο αλγόριθμός μας θα τερματίζει σε χρόνο ίσο με O(m) καθώς τότε θα έχουν εξεταστεί όλες οι ακμές μας. Το αποτέλεσμα θα είναι σωστό εάν έχουν τοποθετηθεί όλες οι κορυφές στο τελικό γράφημα. Εάν, δηλαδή, το πλήθος των κόμβων του τελικού γραφήματος είναι ίσο με V καθώς αν δεν είναι τότε σημαίνει ότι κάποιο από τα $δ_i$ που μας δόθηκαν ήταν μικρότερο του πραγματικού ελαχίστου μονοπατιού. Βέβαια πάντα θα πρέπει να κοιτάω αν το πραγματικό μήκος (όταν ξαναφτάνω σε κόμβο που έχω βάλει στο τελικό γράφημα) είναι μικρότερο από το $δ_ι$ που θα σημαίνει ότι ήταν λάθος το στοιχείο που τοποθετήσαμε εξαρχής και άρα και η σειρά δεν είναι των μικρότερων μονοπατιών.

(β) Εφόσον η μείωση δε δημιουργεί κύκλο αρνητικού μήκους θα θελήσουμε να βρούμε τις νέες αποστάσεις. Θα μειώσουμε όπως μας λέει και η εκφώνηση την ακμή e = (x, y) και θα την κάνουμε w’ από w.

Θα έχω έναν πίνακα με τις αποστάσεις ανά δύο όλων των στοιχείων. Η τροποποίησή μας θα επηρεάσει μόνο τα στοιχεία στα οποία υπάρχει αυτή η ακμή. Γιατί: έστω ότι έχω ένα μονοπάτι p’ που δε διέρχεται από την ακμή e και είναι το ελάχιστο μονοπάτι στο νέο γράφο που δημιουργείται μεταξύ των ακμών (a, b), τότε θα υποθέσω ότι ισχύει πως είναι και το ελάχιστο στον αρχικό μας γράφο και άρα θα ισχύει ότι: d’(a, b) = d(a, b). Αν η υπόθεση αυτή δεν είναι αληθής τότε θα υπάρχει μονοπάτι p στο γράφημα πριν τη μείωση της ακμής θα είναι το ελάχιστου μήκους μεταξύ των (a,b). Τότε θα ισχύει: w’(p) <= w(p) < w(p’) = w’(p’) το οποίο έρχεται σε αντίθεση με τον ορισμό του μονοπατιού p’.

Τώρα θα πρέπει να δούμε πως το βλέπουμε αυτό και πως τα αλλάζουμε τα στοιχεία μας. Θα πάρω την τιμή $w_{i,x}$ + w' + $w_{y,j}$. Αν η τιμή αυτή είναι μικρότερη της τιμής που έχουμε ήδη τότε πάει να πει ότι το μονοπάτι έχει την ακμή που αλλάξαμε και άρα τροποποιούμε το αποτέλεσμα του κελιού που κοιτάμε. Έτσι μετά από n^2 επαναλήψεις, όσες δηλαδή και τα κελιά του πίνακά μας θα έχουμε τα επιθυμητά αποτελέσματα. Οπότε θα έχουμε και πολυπλοκότητα ίση με $O(n^2)$.

(γ) Οι βασικές διαφορές που υπάρχουν είναι πως τώρα δεν μπορούμε να είμαστε σίγουροι αν τα μονοπάτια που περνάνε από την ακμή e είναι τα ελάχιστα που μπορούν να προκύψουν. Αυτό συμβαίνει καθώς μπορεί να προκαλέσει τέτοια αύξηση που να μη γίνεται να την πάρουμε στον αλγόριθμό μας και επομένως να μην μπορούμε γρήγορα να υπολογίσουμε ποιο είναι το μήκος των μονοπατιών από τα ήδη υπολογισμένα μήκη που έχουμε. Δεν μπορούμε δηλαδή να εφαρμόσουμε τη λογική που είπαμε παραπάνω αλλά και καμία άλλη μέθοδο καθώς μπορεί να αλλάξουν ριζικά τα ελάχιστα μονοπάτια.

Άσκηση 2: Σύστημα Ανισοτήτων

(α) Θα φτιάξουμε έναν γράφο που θα έχει σε κάθε κόμβο μια μεταβλητή και μεταξύ των κόμβων i, j την τιμή b_{ij}. Για να δούμε ότι όντως το πρόβλημα μας είναι ικανοποιήσιμο χρειάζεται να εξασφαλίσουμε ότι για κάθε x_i, x_j δεν υπάρχει περίπτωση να δημιουργηθούν ασυνέπειες. Αυτές προκύπτουν μόνο όταν έχουμε $x_i - x_j \leq b_1$ και $x_j - x_i \leq b_2$, όπου b_1, b_2 αρνητικές σταθερές (αν τα b_1, b_2 ήταν θετικά τότε η μία διαφορά που θα είναι αρνητική φράσσεται από κάτι θετικό και άρα ισχύει). Το παραπάνω πρόβλημα στον γράφο μας κωδικοποιείται ως ή μη κύκλων αρνητικού μήκους στον γράφο μας.

Αν υπήρχαν κύκλοι αρνητικού μήκους τότε θα είχαμε τα εξής (έστω για 3 κόμβους):

$x_i - x_j \leq b_1$, $x_j - x_k \leq b_2$, $x_k - x_i \leq b_3$, όπου b_1, b_2, b_3 όλες αρνητικές σταθερές.

Από τις παραπάνω, αν τις προσθέσουμε κατά μέλη προκύπτει ότι $0 \leq b$ όπου b επίσης αρνητική σταθερά, πράγμα που είναι άτοπο. Αν επομένως εξασφαλίσουμε ότι δεν υπάρχουν τέτοια προβλήματα στον γράφο μας το σύστημα S είναι ικανοποιήσιμο. Ο εντοπισμός όμως αρνητικών κύκλων σε γράφο εύκολα εντοπίζεται με τον αλγόριθμο Bellman-Ford και υπολογίζεται σε χρόνο O(n*m). Άρα αρκεί στον παραπάνω γράφο να τρέξουμε μία φορά τον αλγόριθμο αυτό (για τις ανάγκες του αλγορίθμου θα χρειαστεί πιθανώς να προσθέσουμε έναν αρχικό κόμβο, έστω u, χωρίς βάρη ακμών). Εάν εν τέλει δε βρούμε κύκλους αρνητικού μήκους τότε το S είναι ικανοποιήσιμο.

(β) Αν το σύστημα S είναι ικανοποιήσιμο τότε μπορούμε εύκολα να δώσουμε τιμές στις μεταβλητές x_i που έχουμε με τον ακόλουθο τρόπο. Έχουμε χρησιμοποιήσει τον Bellman-Ford που κρατάει του προγόνους οπότε μπορούμε εύκολα να φτιάξουμε ένα δέντρο ελαχίστων μονοπατιών. Έστω ότι το από το δέντρο που έχουμε βάζουμε τη x_1 ίση με μια τιμή, για παράδειγμα 0 για να μη μας επηρεάσει και στη συνέχεια. Έπειτα οι υπόλοιπες

μεταβλητές μπορούν να πάρουν τιμές ίσες με x_1 συν την απόστασή τους από αυτή για να είμαστε σίγουροι ότι όλες οι ανισότητες ικανοποιούνται.
Τώρα, αν το S δεν είναι ικανοποιήσιμο χρειάζεται να υπολογίσω ελάχιστο σύνολο S' που δεν είναι ικανοποιήσιμο. Θα χρησιμοποιήσουμε δυναμικό προγραμματισμό για την εύρεση της λύσης μας. Παίρνουμε έναν πίνακα προγόνων και έναν πίνακα ελαχίστων μονοπατιών. Στον 2º πίνακα (έστω Α) βάζουμε κάθε φορά το συντομότερο μονοπάτι που μας πάει από τον κόμβο i στον κόμβο j κοιτώντας πάντα αν αυτό περνάει από τον κόμβο l ή όχι. Επομένως η αναδρομική σχέση που θα χρησιμοποιήσουμε είναι η:

$A_{\kappa+1}(i, j) = \min\{A_\kappa(i, l) + A_\kappa(l, j), A_\kappa(i, j)\}$, όπου κ ο αριθμός των ακμών.

Στον πίνακα προγόνων βάζουμε -1 αν δεν προσθέσαμε κάποιον καινούργιο κόμβο στο Α αλλιώς l. Εάν πάλι σε κάποιο βήμα μπορώ να υπολογίσω μονοπάτι $A_{\kappa+1} + b_{ij} \leq 0$ τότε υπάρχει κύκλος αρνητικού μήκους που θα έχει το πολύ k+2 ακμές και έπειτα από τον πίνακα των προγόνων μπορώ να εξάγω το S'.
Για τον υπολογισμό του Α κάνω $O(n^3)$ σε κάθε επανάληψη, άρα συνολικά η πολυπλοκότητα του αλγορίθμου είναι $O(n^4)$.

(γ) Στο ερώτημα αυτό θα κάνουμε και πάλι χρήση της προηγούμενης τακτικής με τους 2 πίνακες τροποποιώντας κατάλληλα τις αναδρομικές σχέσεις για να συμπεριλάβουμε και τα βάρη. Φτιάχνουμε πάλι τον πίνακα συντομότερων μονοπατιών με βάση όμως τα βάρη των ακμών. Οπότε ακριβώς αντίστοιχα με πριν έχουμε για το μονοπάτι βάρους k + 1:

$A_{k+1}(i, j) = \min\{A_{k-m}(i, l) + A_{m+1}(l, j), b_{ij} \text{ ,αν } w_{ij} \leq k+1\}$

Επειδή εδώ μας νοιάζουν και τα βάρη μία αντίστοιχη τροποποίηση θα κάνουμε και στον πίνακα των προγόνων, στον οποίο θα βάζουμε (−1, −1) αν δεν κάναμε τροποποίηση στον πίνακα Α, αλλιώς βάζουμε (m, l). Όμοια με πριν εξάγουμε το S' εάν βρούμε αρνητικούς κύκλους. Επομένως έχουμε κόστος $O(n^3)$ για τον υπολογισμό του Α, ενώ για να βρω το S' κάνω μία διάσχιση στον πίνακα προγόνων που πλέον είναι δισδιάστατος και επομένως έχω πολυπλοκότητα $O(n^3 * C^3)$ (ψευδοπολυωνυμικός αλγόριθμος).

(δ) Το πρόβλημα που λύσαμε στο ερώτημα (γ) μπορεί να διατυπωθεί ως πρόβλημα απόφασης ως εξής: Υπάρχει μη ικανοποιήσιμο υποσύστημα S' βάρους μικρότερου ή ίσου με C. Λόγω της δυσκολίας του παραπάνω προβλήματος υποψιαζόμαστε ότι είναι NP-πλήρες. Για να το αποδείξουμε ωστόσο πρέπει να κάνουμε αναγωγή σε κάποιο γνωστό NP-πλήρες πρόβλημα. Επιλέγουμε το πρόβλημα του διακριτού σακιδίου. Έστω σακίδιο μεγέθους B και n στοιχεία με βάρη w_ι και κέρδος p_ι. Στόχος είναι να βρούμε ένα υποσύνολο των n στοιχείων τέτοιο ώστε να χωράει στο σακίδιο (συνολικό βάρος μικρότερο ή ίσο του B) και αξία μεγαλύτερη ή ίση ενός P. Κάθε στοιχείο, έστω x_ι, το αναπαριστούμε με ένα κόμβο. Η ακμή που θα συνδέει τα x_ι, $x_\iota - 1$ θα πρέπει να έχει βάρος $P_{o\lambda} - p_\iota$ και μήκους w_ι και με μια ακμή βάρους $P_{o\lambda}$ και μήκος 0.
Θεωρούμε μια αρχική κορυφή q_0 και στην τελευταία κορυφή βάζω μια ακμή από αυτή στην πρώτη με βάρος 0 και μήκος −(B+1). Αν σε αυτό το γράφημα βρω ένα κύκλο

αρνητικού μήκους τότε στο πρόβλημα του σακιδίου θα έχω κάνει μια επιλογή στοιχείων με συνολικό βάρος μικρότερο ή ίσο του B. Σε αυτό τον κύκλο θα μετράνε μόνο οι ακμές του πρώτου είδους. Για να βρω ένα υποσύνολο συνολικού βάρους μικρότερο ή ίσο από το $n*P_{ολ}-P$. Το τελευταίο λύνει το γενικό πρόβλημα του σακιδίου και είναι NP-Complete άρα το πρόβλημα μας είναι τουλάχιστον της ίδιας δυσκολίας. Αν μας δοθεί μια λύση τότε μπορούμε σε πολυωνυμικό χρόνο να βρούμε αν το υποσύνολο έχει κύκλο αρνητικού μήκους και βάρος μικρότερου ή ίσο του C. Άρα το πρόβλημα ανήκει στο NP και αφού κάναμε αναγωγή του προβλήματος του σακιδίου σε αυτό είναι και NP-Complete.

Άσκηση 3: Ταξιδεύοντας με Ηλεκτρικό Αυτοκίνητο

(α) Έχουμε ότι |C| >= 3. Αυτό που είναι σημαντικό είναι να περάσουμε μεταξύ μιας από τις έξτρα πόλεις με σταθμό φόρτισης. Αν είναι μόνο 3 πόλεις, τότε χρειάζεται να βρούμε την ελάχιστη απόσταση που να περνά από την πόλη s στην ενδιάμεση πόλη τροφοδότησης και από εκεί στην τελική πόλη t. Τώρα εάν έχουμε πολλές ενδιάμεσες πόλεις χρειάζεται να βρούμε ποια είναι η ελάχιστη απόσταση που έχουμε από το s στο t και να περνάμε από τουλάχιστον μία από αυτές. Χρησιμοποιούμε Dijkstra για να υπολογίσουμε τις αποστάσεις από τον κόμβο s προς όλους τους ενδιάμεσους κόμβους και από τους κόμβους αυτούς προς τον t. Από όλες τις αποστάσεις που θα βρούμε κρατάμε αυτές με το μικρότερο άθροισμα. Έτσι θα έχουμε χρόνο O(C*E*logV) καθώς και οι δύο υπολογισμοί προκύπτουν από το ίδιο χρόνο για να μπορέσουμε να τους βρούμε.

(β) Τώρα εφόσον το ντεπόζιτο δε μας φτάνει για την απευθείας μετακίνηση θα εργαστούμε ως εξής. Θα χρησιμοποιήσουμε αλγόριθμό Dijkstra μεταξύ από όλους τους κόμβους που ανήκουν στο C με τον περιορισμό πως αν μια από τις ακμές είναι μεγαλύτερη του ντεπόζιτου που έχουμε τότε δεν την κοιτάμε καν. Έτσι θα έχουμε έναν νέο γράφο, ο οποίος θα περιέχει όλους τους κόμβους που ανήκουν στο C με τα μήκη για να πάμε από τον έναν στον άλλο, εάν υπάρχει βέβαια τέτοιο μονοπάτι. Οι αποστάσεις, δηλαδή, που θα υπάρχουν είναι οι ελάχιστες αποστάσεις μεταξύ των κόμβων τροφοδότησης. Οπότε, το ζητούμενό μας είναι να τρέξουμε μία ακόμα φορά τον αλγόριθμο του Dijkstra και να βρούμε το μήκος του ελαχίστου μονοπατιού από τον κόμβο s προς τον κόμβο t. Αν πάλι τα μήκη είναι μεγαλύτερα της αυτονομίας του αυτοκινήτου μας, τότε απλά μπορούμε να μην τα προσθέτουμε στο νέο μας γράφημα ώστε αυτή τη φορά να τρέξουμε χωρίς περιορισμούς τον αλγόριθμό μας.

Η πολυπλοκότητα του αλγορίθμου μας είναι O(C*E*logV) καθώς το $1^{ο}$ σκέλος του αλγορίθμου μας τρέχει σε αυτό τον χρόνο και στο επόμενο έχουμε C κόμβους για να κάνουμε τον Dijkstra και άρα θα είναι πιο γρήγορο για την ακρίβεια O(C*C*logC), εφόσον μπορούμε να έχουμε το πολύ C-1 ακμές στις C κορυφές που έχει το γράφημά μας.

Άσκηση 4: Επιβεβαίωση και Αναπροσαρμογή Μέγιστης Ροής.

(α) Ο τρόπος που έχουμε για να δούμε εάν η ροή f που μας έχει δοθεί είναι η μέγιστη είναι ο υπολογισμός του υπολειμματικού γράφου. Εάν όντως η ροή που μας δίνεται είναι η μέγιστη τότε δε θα μπορούμε να βρούμε μονοπάτι το οποίο συνδέει τον κόμβο s με τον κόμβο t. Αυτό συμβαίνει γιατί στις μέγιστες ροές έχουν ικανοποιηθεί πλήρως ορισμένες από τις ακμές που ενώνουν το s με το t και επομένως δεν μπορούμε να οδηγηθούμε από το ένα στο επόμενο. Εάν μπορούσαμε πάει να πει ότι το σύστημά μας μπορεί να αυξήσει τη ροή σε κάποια από τις ακμές και άρα και η τελική ροή να βγει μεγαλύτερη. Αυτό συμβαίνει γιατί ο τρόπος τοποθέτησης μιας ακμής (α,β) στον υπολειμματικό γράφο είναι ίσος με τη χωρητικότητα-ροή αν υπάρχει η ακμή από τον (α,β) στο αρχικό μας γράφημα και ίσο με τη ροή αν είναι η ακμή (β,α) στο αρχικό μας γράφημα. Έτσι εάν μια ακμή είναι 0 στο τέλος (χρησιμοποιείται όλη η χωρητικότητα της γραμμής) δε θα μπορεί να υπάρξει συνοχή μεταξύ αρχής και τέλους. Για την εύρεση της συνοχής χρειαζόμαστε έναν αλγόριθμο BFS ή DFS. Οπότε τελικά έχουμε γραμμικό χρόνο μιας και οι χρειαζόμαστε γραμμικό χρόνο για την κατασκευή του γραφήματος και γραμμικό χρόνο για να δούμε αν υπάρχει το μονοπάτι s-t.

(β) Καταρχάς αν η μέγιστη ροή στην ακμή που μειώσαμε ήταν εξαρχής 0 τότε δεν χρειάζεται να αλλάξω κάτι γιατί θα παραμείνει μηδέν.
Τώρα αν δεν ισχύει αυτό, αλλάζουμε τη ροή σε όλες τις ακμές από f σε f’. Για όλες τις ακμές πέρα από αυτή που μειώθηκε ισχύει ότι f = f’ και για την ακμή που μειώσαμε ισχύει ότι f’ = f − k. Παρόλο που ο περιορισμός χωρητικότητας ικανοποιείται μετά τη μείωση της ροής, ο περιορισμός της διατήρησης ροής παραβιάζεται στους 2 κόμβους - έστω u, v- που συνδέονται από την ακμή που το μέγεθος της έχει αλλάξει. Η f’ έχει k μονάδες περισσότερες που μπαίνουν στο u από ότι βγαίνουν από αυτό και έχει k περισσότερες μονάδες που φεύγουν από το v απ’ ότι μπαίνουν στο v. Προκειμένου να εξισορροπήσουμε αυτό το σφάλμα που υπάρχει χρειάζεται να προσπαθήσουμε να ανακατασκευάσουμε αυτές τις k μονάδες ροής μέσω κάποιου άλλου μονοπατιού. Αν δεν είναι δυνατόν αυτό θα πρέπει να μειώσουμε τη ροή από το s στο u και από το v στο t κατά k μονάδες. Ψάχνουμε για ένα επαυξητικό μονοπάτι από το u στο v. Αν βρούμε τέτοιο μονοπάτι τότε μειώνουμε τη ροή από το s στο u μειώνοντας τη ροή από το u στο s. Αυτό θα το κάνουμε βρίσκοντας ένα επαυξητικό μονοπάτι από το u στο s και αυξάνοντας τη ροή του. Κάτι αντίστοιχο κάνουμε και στο v, t. Όμως αυτή η αναδιανομή της ροής που περιγράψαμε πιο πάνω δεν θα δίνει και για τις k μονάδες εξαρχής αλλά θα γίνει για κάθε μια μονάδα ξεχωριστά. Αν δούμε ότι σε κάποια μονάδα δεν μπορούμε να αναδιανείμουμε τη ροή τότε προσπαθούμε να κάνουμε το δεύτερο βήμα, δηλαδή να μειώσουμε την ελάχιστη ροή Η πολυπλοκότητα του αλγορίθμου μας είναι ίση με αυτή του BFS/DFS για να βρούμε τα μονοπάτι δηλαδή ίση με O(V + E) = O(E)

(γ) Αρχικά για κάθε ακμή θα κάνουμε την αφαίρεση $f_e - l_e$ για να βρούμε πόσο μπορεί να μειωθεί η ροή σε κάθε ακμή. Στη συνέχεια θα έχουμε το πρόβλημα μέγιστης ροής αλλά αντίστροφα. Δηλαδή αντί να ψάχνουμε η μέγιστη ροή από το s στο t θα ψάχνουμε τη μέγιστη ροή από το t στο s. Αφού χρησιμοποιήσουμε τον αλγόριθμο του Ford-Fulkerson για να υπολογίσουμε αυτή τη ροή στη συνέχεια αφαιρούμε από την πρώτη τη μέγιστη ροή σε κάθε ακμή τη δεύτερη μέγιστη ροή. Με αυτό τον τρόπο βρίσκουμε τη ροή που θα θέλαμε να μας μένει. Η πολυπλοκότητα του αλγορίθμου μας είναι O(E*max|f|), όπου max|f| είναι η μέγιστη ροή που είχαμε, όσο δηλαδή και η πολυπλοκότητα του Ford-Fulkerson. Βέβαια αν θέλουμε να μην εξαρτάται από τη ροή μπορούμε να χρησιμοποιήσουμε κάποιον από τους αλγορίθμους Edmonds-Karp, Dinic's blocking flow, MPM, Dinic's αλγόριθμους οι οποίοι υπολογίζουν τη μέγιστη ροή σε χρόνο $O(V*E^2)$, $O(V^2*E)$, $O(V^3)$ O(V*E*logV) αντίστοιχα ή του James B Orlin's + KRT αλγόριθμους που ο καθένας λύνουν το πρόβλημα σε O(V*E) με περιορισμούς στο πλήθος των ακμών σε σχέση με το πλήθος των κόμβων.

Άσκηση 5: Αναγωγές και NP-Πληρότητα

- 3-Διαμέριση:

Το πρόβλημα της διαμέρισης είναι όταν έχουμε N στοιχεία με βάρη w_i να τα χωρίσουμε 2 ισοβαρή υποσύνολα. Το πρόβλημα αυτό είναι γνωστό ότι ανήκει στην κλάση NP-hard, οπότε θα προσπαθήσουμε να δείξουμε ότι αυτό ανάγεται στο 3-διαμέριση. ΄
Έστω ότι μας δίνονται N στοιχεία με βάρη w_i και ότι καταφέρνουμε να τα χωρίσουμε δύο ισοβαρή υποσύνολα. Αν στο αρχικό σύνολο προσθέσουμε μία επιπλέον ποσότητα ίση με το μισό του συνολικού βάρους (όσο δηλαδή ζυγίζουν και τα 2 υποσύνολα που χωρίσαμε) τότε έχουμε καταφέρει να χωρίσουμε ένα σύνολο N στοιχείων σε 3 ισοβαρή υποσύνολα. Άρα το πρόβλημα διαμέρισης ανάγεται στο 3- Διαμέριση και επομένως είναι NP-hard. Από την άλλη εάν έχουμε μία διαμέριση μπορούμε σε πολυωνυμικό χρόνο να δούμε εάν είναι 3-Διαμέριση, οπότε το πρόβλημα είναι NP-Complete.

- Μακρύ μονοπάτι:

Προσθέτουμε σε κάθε κορυφή γείτονες βαθμού 1. Αν έχουμε μονοπάτι Hamilton στο G, τότε θα υπάρχει μονοπάτι μήκους |V|+2 στο G′. Αυτό που θέλουμε στο τέλος εμείς είναι το μήκος του μονοπατιού να είναι τουλάχιστον το 1/4 του συνολικού αριθμού των κορυφών του G′. Οπότε χρειάζεται τελικά να προσθέσουμε αρκετές καινούργιες κορυφές ώστε να έχουμε |V'| = 4|V|+8. Η εύρεση του μονοπατιού Hamilton είναι NP-complete οπότε και το ζητούμενό μας θα είναι NP-complete. Το μονοπάτι Hamilton μπορούμε να πούμε ότι είναι NP-complete καθώς μπορούμε να το ανάγουμε στον κύκλο Hamilton που γνωρίζουμε πως είναι και αυτός NP-complete.

- Πυκνό Γράφημα (Dense Subgraph):

Μπορούμε να δούμε ότι το πρόβλημά μας είναι μια γενίκευση της κλίκας με $b = k(k-1)/2$ επομένως είναι πολύ απλό το να πούμε πως θα είναι και NP-complete όπως και το πρόβλημα το οποίο είναι η εξειδίκευσή του. Αν έχουμε ένα σύνολο S μπορούμε εύκολα να δούμε εάν είναι πυκνό γράφημα όπως θέλουμε.

- Ικανοποιησιμότητα με Περιορισμούς:

Είναι μια γενίκευση του SAT προβλήματος. Θα πρέπει να μπορούν να ικανοποιηθούν τα m υποπροβλήματα του φ με τον περιορισμό ότι έχουμε τουλάχιστον 1 αληθές και ένα ψευδές literal. Επομένως είναι και NP-complete το πρόβλημά μας καθώς το ανάγουμε στο SAT.

- Επιλογή Ανεξαρτήτων Υποσυνόλων:

Είναι μια γενίκευση του μονοπατιού Hamilton. Αυτό που θέλουμε εδώ είναι να μην υπάρχουν μονοπάτια μεταξύ των k υποσυνόλων, καθώς τότε θα είναι ανά δύο ξένα μεταξύ τους. Γι’ αυτό και το πρόβλημά μας θα είναι NP-complete.