

Domination on modular product graphs

Poročilo

Domen Humar in Maja Komic

November 2023

1 Definicije

Definicija 1 Podmnožica vozlišč grafa G (označimo jo z S) je **dominanta podmnožica** grafa G , če zanjo velja, da je vsako vozlišče grafa G , ali znotraj podmnožice S , ali je sosednjo nekemu vozlišču znotraj S .

Definicija 2 **Dominacijsko število** grafa G , označeno z $\gamma(G)$, je velikost/moč/kardinalno število najmanjše dominantne podmnožice grafa G .

Definicija 3 **Modularni produkt grafov G in H** je graf $G \diamond H$ z množico vozlišč $V(G \diamond H) = V(G) \times V(H)$, ki je unija kartezičnega produkta, direktnega produkta in direktnega produkta komplementov G in H

$$G \diamond H = G \square H \cup G \times H \cup \overline{G} \times \overline{H}$$

. Natančneje, točki (g, h) in (g', h') iz grafa $G \diamond H$ sta sosednji, če velja:

1. če je $g = g'$ in $hh' \in E(H)$; ali
2. če je $h = h'$ in $gg' \in E(G)$; ali
3. če je $gg' \in E(G)$ in $hh' \in E(H)$; ali
4. če za $g \neq g'$ in $h \neq h'$ velja $uu' \notin E(G)$ in $hh' \notin E(H)$.

Povezave iz prve in druge točke so iz kartezičnega produkta, povezave iz tretje točke so iz direktnega produkta in povezave iz četrte točke so iz direktnega produkta komplementov.

2 Problem

Naj bosta G in H grafa. Na različnih primerih grafov želimo preveriti spodnjo neenakost in poiskati čim več takih grafov G in H za katera velja enakost.

$$\gamma(G \diamond H) \leq \gamma(G) + \gamma(H) - 1 \quad (1)$$

Primer 1 Naj bo G graf z 4 vozlišči in 3 povezavami in H naj bo graf s 3 vozlišči in 2 povezavama. Dominacijsko število grafa G je 2, dominacijsko število grafa H pa 1. Ko naredimo modularni produkt na grafih G in H , dobimo graf $G \diamond H$, ki ima dominacijsko število 2.

$$\begin{aligned} \gamma(G \diamond H) &\leq \gamma(G) + \gamma(H) - 1 \\ 2 &\leq 2 + 1 - 1 \end{aligned}$$

Vidimo, da velja neenakost in enakost.

3 Reševanje problema

Za reševanje opisanega problema sva uporabila programski jezik *Python* in okolje Sage (SageMath). Najprej sva s pomočjo zgornjih definicij definirala funkciji *dominacijsko_stevilo*(G, H) in *dominacijsko_stevilo*(G). Ter funkciji, ki preverjata neenakost (1) oz enakost.

```
# MODULARNI PRODUKT
def modularni_produkt(G, H):
    A = H.cartesian_product(G)
    B = H.tensor_product(G)
    C = H.complement().tensor_product(G.complement())
    X = A.union(B)
    Y = X.union(C)
    return Y

# DOMINACIJSKO ŠTEVILO
def dominacijsko_stevilo(G):
    p = MixedIntegerLinearProgram(maximization = False)
    b = p.new_variable(binary = True)
    p.set_objective( sum([b[v] for v in G]) )
    for u in G:
        p.add_constraint(
```

```

        b[u] + sum([b[v] for v in G.neighbors(u)]) >= 1
    )
    a = p.solve()
    return a

# FUNKCIJA, KI PREVERI NEENAKOST
def preveri_neenakost(x, y, z):
    return x <= y + z - 1

# FUNKCIJA, KI PREVERI ENAKOST
def preveri_enakost(x, y, z):
    return x == y + z - 1

```

Nato sva z uporabo zgornjih funkcij izvedla simulacijo, ki na parih grafov z raznimi kombinacijami števila vozlišč in povezav, poišče modularni produkt grafa, izračuna dominacijska števila osnovnih grafov in njunega modularnega produkta, ter preveri neenakost in enakost. Funkcija *naredi_matriko*(*n*, *m*) sprejme parametra *n* in *m*, ki označujeta največje število vozlišč grafov *G* in *H*. Vne pa matriko, ki ima v vrstici podatke število vozlišč grafa *G*, število povezav grafa *G*, dominacijsko število grafa *G*, število vozlišč grafa *H*, število povezav grafa *H*, dominacijsko število grafa *H*, dominacijsko število modularnega produkta grafov *G* in *H*, True, če neenakost (1) velja oziroma False, če neenakost (1), ter True če velja enakost oziroma False, če enakost ne velja.

```

# SIMULACIJA
def naredi_matriko(n, m):
    for k in range(2, n):
        for H in graphs(k):
            for d in range(2, m):
                for G in graphs(d):
                    Y = modularni_produkt(G, H)
                    y = dominacijsko_stevilo(Y)
                    h = dominacijsko_stevilo(H)
                    g = dominacijsko_stevilo(G)
                    i = preveri_neenakost(y, h, g)
                    j = preveri_enakost(y, h, g)
                    matrika.append([H.order(), H.size(), h,
                                     G.order(), G.size(), g, y, i, j])
    return matrika

```

4 Rezultati

Z izvedbo zgornjega algoritma z grafi do vključno 5 vozlišč sva dobila tabelo, ki je v Prilogi 1, iz katere je razvidno, da neenakost (1) velja za vse grafe. S podrobnejšo analizo tabele iz priloge 1 pa sva prišla do zaključka, da enakost nastopi v primeru, ko je dominacijsko število enega od grafov natanko 1.