توضیحات پروژه سوم درس هوش مصنوعی

روال کار بدین صورت است که درابتدا، ورودی های لازم(تعداد راس، تعداد یال، شکل راس ها و مشخصات یال ها) را دریافت می کند و یک گراف که دامنه ی راس های آن، اعداد یک رقمی می باشد، می سازد. با توجه به استفاده از کتابخانه networkx، این گراف قابلیت به نمایش در آمدن نیز دارد.

```
print('Enter vertices: ')
vertice_string = input().split()
for i in range(len(vertice_string)):
    nodes.append(vertice_string[i])

print('Enter edges: ')
for i in range(e_size):
    edge = []
    edge_string = input().split()
    edge.append(int(edge_string[0]))
    edge.append(int(edge_string[1]))
    edges.append(edge)

G = nx.Graph()

#graph vertices
for i in range(len(nodes)):
    if nodes[i] == 'T':
        G.add_node(i, shape = '^', label = '', domain = [1, 2, 3, 4, 5, 6, 7, 8, 9])
    elif nodes[i] == 'S':
        G.add_node(i, shape = 's', label = '', domain = [1, 2, 3, 4, 5, 6, 7, 8, 9])
    elif nodes[i] == 'P':
        G.add_node(i, shape = 'p', label = '', domain = [1, 2, 3, 4, 5, 6, 7, 8, 9])
    elif nodes[i] == 'H':
        G.add_node(i, shape = 'h', label = '', domain = [1, 2, 3, 4, 5, 6, 7, 8, 9])
    elif nodes[i] == 'C':
        G.add_node(i, shape = 'o', label = '', domain = [1, 2, 3, 4, 5, 6, 7, 8, 9])
```

حال به انتخاب کاربر، یکی از دو حالت جستجوی عقب گرد با وارسی پیش رو(bt+fc) یا جستجوی عقب گرد با وارسی پیش رو و هیوریستیک mrv (bt+fc+mrv) انتخاب می شود.

```
for i in range(1):
    print('Select 1 for BT-FC or 2 for BT-FC-MRV')
    mode = input()
    if(mode != '1' and mode != '2'):
        print('Please try again!')
        i = i - 1
mode_int = int(mode)


if mode_int == 1:
    bt_fc(G, 1) #incremental selection
if mode_int == 2:
    bt_fc(G, 2) #MRV selection
```

سپس می آییم و مادامی که تابع finished، true نشده است(در صورتی true می شود که همه ی متغیرها، مقدار داشته باشند) و تابع failed نیز true نشده باشد(در صورتی true می شود که دامنه ی همه ی متغیرها تهی شده باشند)، آنگاه یک متغیر را طبق سیاست خواسته شده(به ترتیب شماره یا هیوریستیک MRV) انتخاب می شود.

پس از انتخاب نود، می آییم و مقادیر دامنه اش را به آن می دهیم و وارسی پیشرو را انجام می دهیم(که تابع check_consistency می باشد)

اگر این نود به ازای جمیع مقادیر دامنه، نتواند مقداری بگیرد، آن گاه عقب گرد می کنیم.

عقب گرد بدین صورت است که دامنه ی آن نود را restore می کنیم؛ سپس از لیست assignment_queue(که شامل متغیرهایی است که مقداردهی شده اند)، آخرین متغیری که مقدار دارد را برمی داریم و آن مقدار را از دامنه اش حذف می کنیم و مقدار فعلیش را نیز حذف می کنیم. سپس به روال طبیعی جستجو ادامه می دهیم.

اگر در حین pop کردن از assignment_queue، این لیست خالی شود، بدین معناست که دیگر نمی توانیم عقب گرد کنیم و مسئله جواب ندارد. بنابراین، جستجو متوقف شده و پیام "ناموفقیت آمیز" برای کاربر چاپ می شود.

تست کیس های پروژه با جوابی که برمی گردانند(یا بدون جواب هستند!)، در فایل test.txt قرار دارد.

نمونه ای از اجرای برنامه:

سیدمتین توکلی افشاری 9631805
پاییز 98