lab3_Augustyniak_Alicja_biostrukturRNA

May 5, 2025

OCENA 4.0 : wyszukiwanie motywów: GNRA, UNCG, T-loop

[32]:
```python
import json
import sys
from pathlib import Path
import argparse


def parse_pattern(pat: str):
    listt = [ch for ch in pat if ch in '().']
    opens = [i for i, ch in enumerate(listt, start=1) if ch == '(']
    closes = [i for i, ch in enumerate(listt, start=1) if ch == ')']
    return listt, opens, closes


def load_data(json_path: Path):
    data = json.loads(json_path.read_text())
    id_to_index = {}
    index_to_code = {}
    for nt in data.get('nts', []):
        idx = nt.get('index')
        id_to_index[nt.get('nt_id')] = idx
        index_to_code[idx] = nt.get('nt_code')
    pairs = {}
    for p in data.get('pairs', []):
        i = id_to_index.get(p.get('nt1'))
        j = id_to_index.get(p.get('nt2'))
        if i is None or j is None:
            continue
        a, b = sorted((i, j))
        pairs[(a, b)] = p.get('LW')
    return index_to_code, pairs


def pair(pairs: dict, i: int, j: int, lw_expected: str) -> bool:
    a, b = sorted((i, j))
    return pairs.get((a, b)) == lw_expected
```

```python
def find_motifs(index_to_code: dict, pairs: dict):
    motifs = {
        'GNRA': [('cWW', '(....)'), ('tSH', '.(..).')],
        'UNCG': [('cWW', '(....)'), ('tSW', '.(..).')],
        'T-loop': [('cWW', '(......)'), ('tSH', '.(...)..')],
    }
    found = []
    for name, rows in motifs.items():
        lw0, pat0 = rows[0]
        list0, opens0, closes0 = parse_pattern(pat0)
        span = len(list0)
        for (i, j), lw in pairs.items():
            if lw != lw0:
                continue
            if (j - i + 1) != span:
                continue
            good = True
            for k in range(len(opens0)):
                a = i + opens0[k] - 1
                b = i + closes0[-(k+1)] - 1
                if not pair(pairs, a, b, lw0):
                    good = False
                    break
            if not good:
                continue
            for lwk, patk in rows[1:]:
                listk, opensk, closesk = parse_pattern(patk)
                if not opensk or not closesk:
                    good = False
                    break
                a = i + opensk[0] - 1
                b = i + closesk[-1] - 1
                if not pair(pairs, a, b, lwk):
                    good = False
                    break
            if not good:
                continue
            if len(opens0) == 1:
                seq = ''.join(index_to_code.get(k, 'N') for k in range(i, j+1))
                found.append((name, [(i, seq, j)]))
            else:
                inner_open = opens0[1]
                inner_close = closes0[-2]
                a2 = i + inner_open - 1
                b2 = i + inner_close - 1
                left_seq = ''.join(index_to_code.get(k, 'N') for k in range(i,␣
↪a2+1))
                right_seq = ''.join(index_to_code.get(k, 'N') for k in␣
↪range(b2, j+1))
                found.append((name, [(i, left_seq, a2), (b2, right_seq, j)]))
    return found


def wresults(json_path: Path, motifs):
    out_path = json_path.with_name(json_path.stem + '_motifs.txt')
    with out_path.open('w') as w:
        if motifs:
            for name, segs in motifs:
                w.write(f"{name}:\n")
                if len(segs) == 1:
                    start, seq, end = segs[0]
                    w.write(f"  {start} {seq} {end}\n")
                else:
                    l = segs[0]
                    r = segs[1]
                    w.write(f"  {l[0]} {l[1]} {l[2]} & {r[0]} {r[1]} {r[2]}\n")
        else:
            w.write("nie znaleziono\n")
    print(f"zapisano: {out_path}")



def main():
    target = None
    for arg in sys.argv[1:]:
        if not arg.startswith('-') and Path(arg).exists():
            target = Path(arg)
            break
    if target is None:
        target = Path('.')
    if target.is_dir():
        json_list = list(target.glob('*.json'))
    else:
        json_list = [target]
    for json_file in json_list:
        index_to_code, pairs = load_data(json_file)
        motifs = find_motifs(index_to_code, pairs)
        wresults(json_file, motifs)


if __name__ == '__main__':
    main()
```

```
zapisano: 1s72_motifs.txt
zapisano: 4ioa_motifs.txt
zapisano: 1vy4_motifs.txt
zapisano: 1byj_motifs.txt
zapisano: 4v54_motifs.txt
zapisano: 4v8p_motifs.txt
```