Programming Exercise 1
Construction Heuristic

Borna Feldsar
Soeren Nickel

Approach

1. Preprocessing
    a. Battery constraints
    b. Time constraints
    c. Capacity constraints
2. Parallel Savings
    a. Initial route creation
    b. Calculate the savings
        i. customer - customer
        ii. charger - customer
    c. Delete charger only routes
    d. Combine routes
3. Verify/Visualize

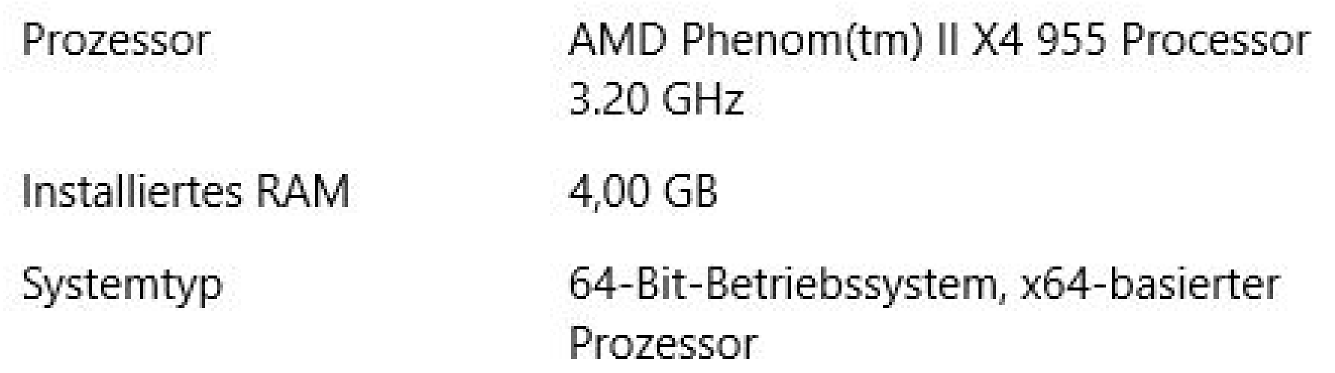
Used Ressources

- Provided paper (The Electric Vehicle-Routing Problem with Time Windows and Recharging Stations; Schneider, Stenger, Goeke) and slides
- Python 3.x
- Provided Verifier
- Graphviz/Dot for visualization

Results

- Total Runtime: 206.125036 seconds
- Average Cost over all Instances: 1065.25
- Worst Cost: instance: ../data/instances/rc101_21.txt
  cost: 3115.0410853411863
  time: 3.442202091217041
- Worst Runtime: instance: ../data/instances/r112_21.txt
  cost: 1344.055276592745
  time: 9.748563051223755

Computer



Prozessor	AMD Phenom(tm) II X4 955 Processor 3.20 GHz
Installiertes RAM	4,00 GB
Systemtyp	64-Bit-Betriebssystem, x64-basierter Prozessor

Preprocessing

q_i - demand of vertex *i*
C - vehicle load capacity
e_i - beginning of a time window at vertex i
l_i - end of a time window at vertex *i*
Q - vehicle battery capacity

Arc (*v,w)* is blacklisted if:

$v,w \in V: q_v + q_w > C$

$v \in V_d , w \in V_d: e_v + s_v + t_{vw} > l_w$

$v \in V_d, w \in V_d: e_v + s_v + t_{vw} + s_w + t_{wd} > l_d$

$v,w \in V, \forall j \in F_d, i \in F_d: h^*(d_{jv} + d_{vw} + d_{wi}) > Q$

Parallel Savings

Initial Routes:

- initial routes are created for every node
- customer might not be reachable without charger
- insert charger after customer and, if needed, before customer

Parallel Savings

Calculate Savings:

- Calculate savings between:
  a. Customer - Customer
  b. Charger - Customer
  c. Both not on Blacklist

Parallel Savings

Delete Charger only Routes

- delete routes that only contain charger
- adding chargers is not done by savings
- add instead nearest charger by necessity

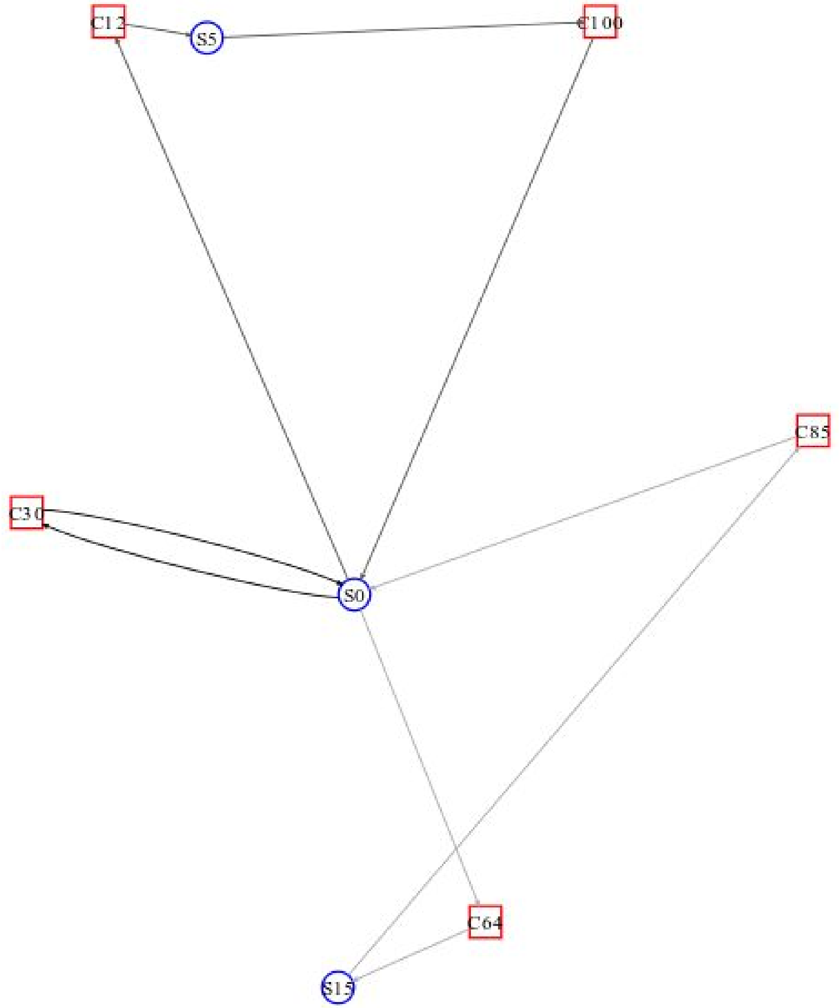
Parallel Savings

Combine Routes

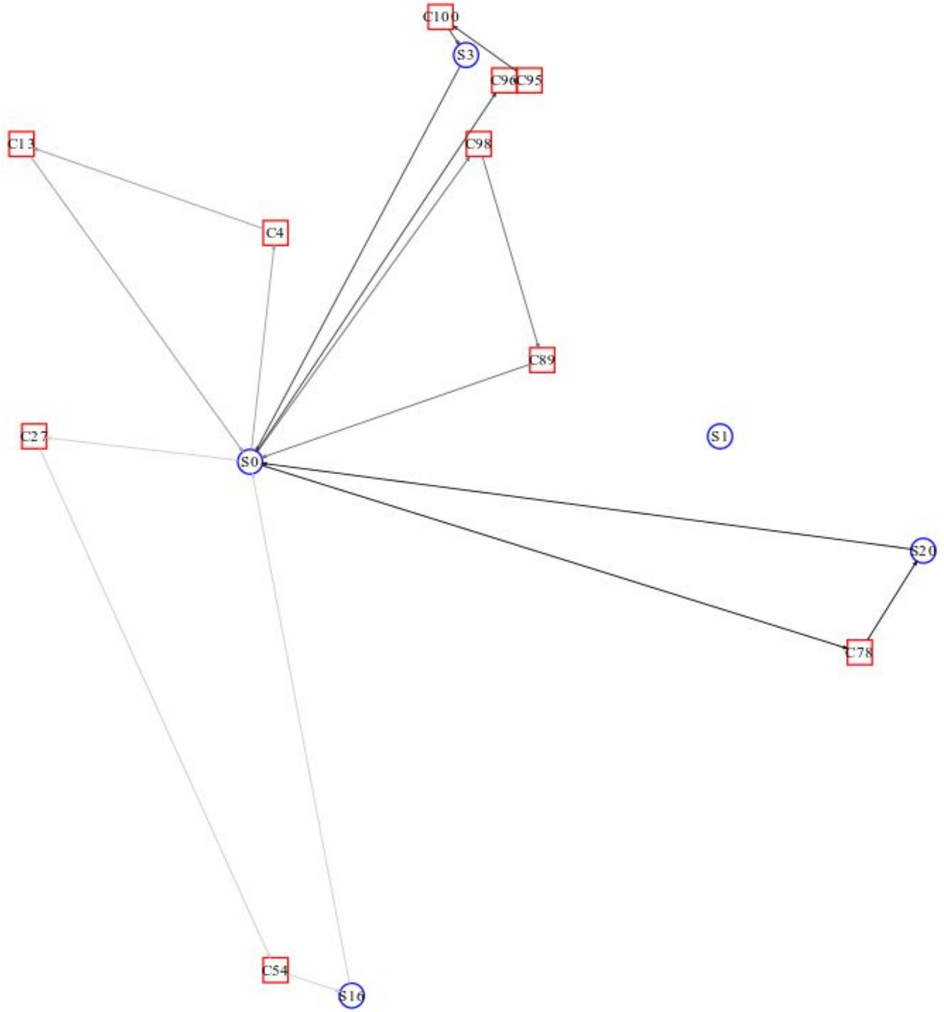
- Create new combined Route
- Check if and where charge is exceeded
    - iterate from there backwards and insert nearest charger if possible
    - if no previous insertion makes combined routes feasible then insert the closest charger between two farthest nodes
    - since we insert chargers until we make our route feasible, we have to decline routes that have more than some number of chargers, because we can create an infinite loop
- Check capacity, time-windows and charge capacity
- If newly created Route is feasible, add it and remove the old ones
- At the end, check for one edge case

Result Examples

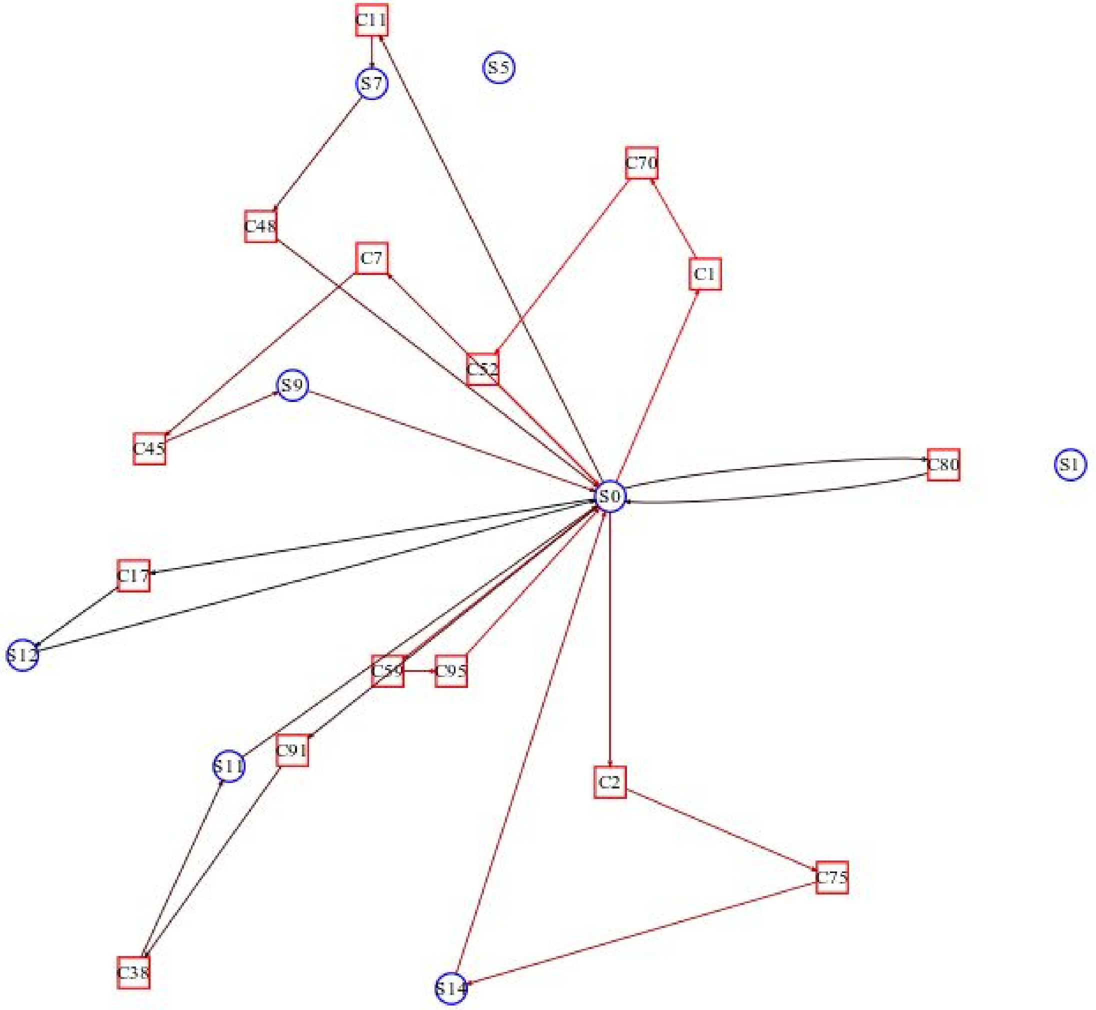
c101C5



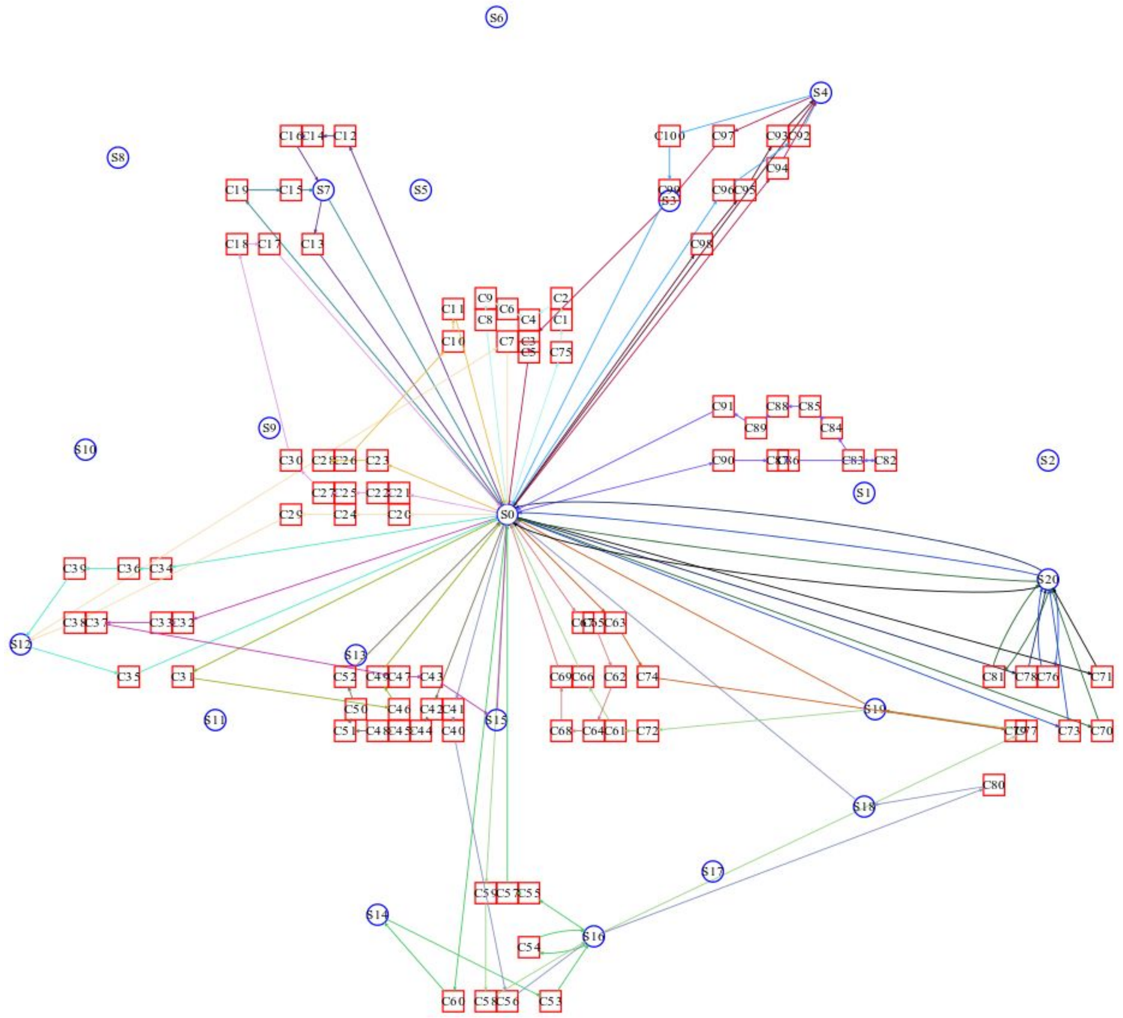
c101C10



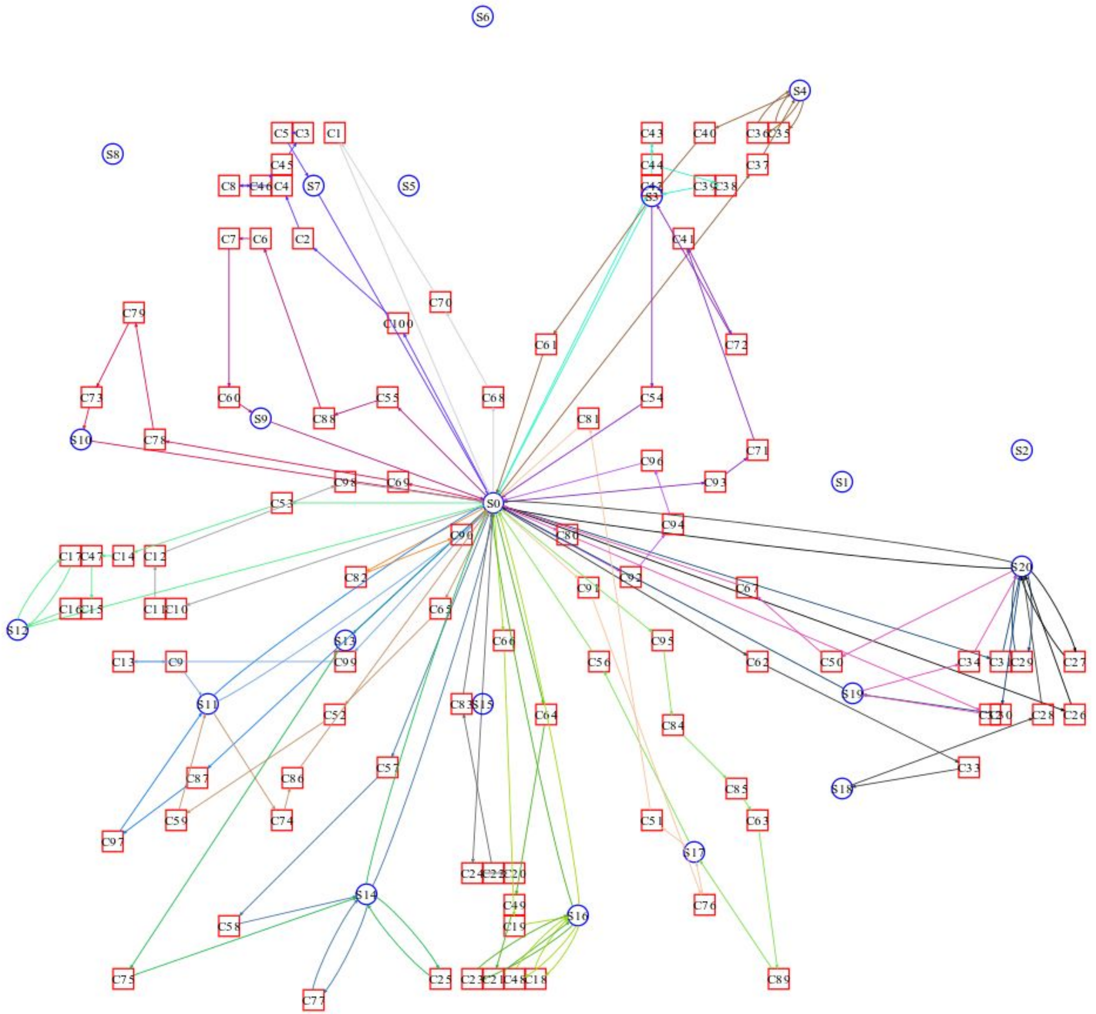
rc102C15


C11
S5
S7
C70
C48
C7
C1
C52
S9
C45
C80
S1
S0
C17
S12
C59
C95
C91
S11
C2
C75
C38
S14

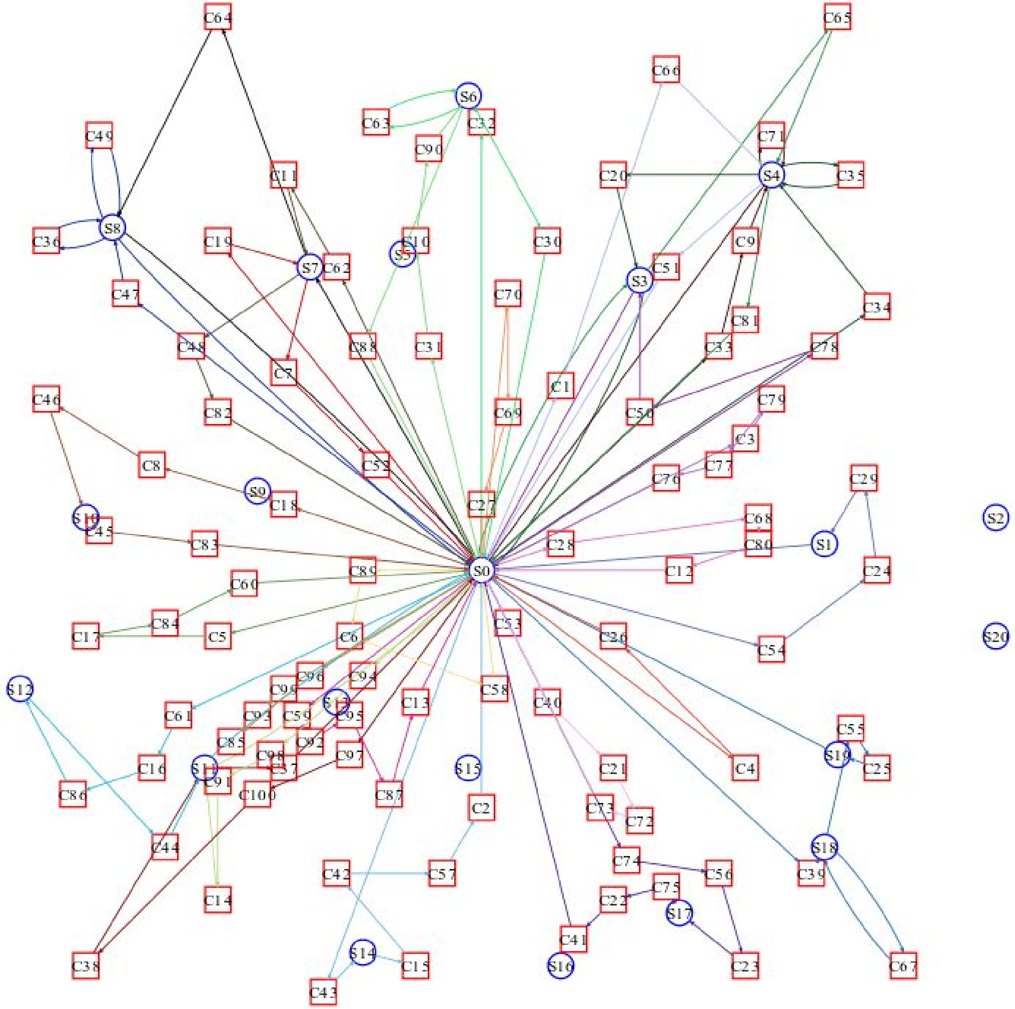
c101_21


S0
S1
S2
S3
S4
S5
S6
S7
S8
S9
S10
S11
S12
S13
S14
S15
S16
S17
S18
S19
S20
C1
C2
C3
C4
C5
C6
C7
C8
C9
C10
C11
C12
C13
C14
C15
C16
C17
C18
C19
C20
C21
C22
C23
C24
C25
C26
C27
C28
C29
C30
C31
C32
C33
C34
C35
C36
C37
C38
C39
C40
C41
C42
C43
C44
C45
C46
C47
C48
C49
C50
C51
C52
C53
C54
C55
C56
C57
C58
C59
C60
C61
C62
C63
C64
C65
C66
C67
C68
C69
C70
C71
C72
C73
C74
C75
C76
C77
C78
C79
C80
C81
C82
C83
C84
C85
C86
C87
C88
C89
C90
C91
C92
C93
C94
C95
C96
C97
C98
C99
C100

rc105_21


S0
S1
S2
S3
S4
S5
S6
S7
S8
S9
S10
S11
S12
S13
S14
S15
S16
S17
S18
S19
S20
C1
C2
C3
C5
C100
C70
C68
C61
C54
C72
C71
C93
C96
C94
C81
C80
C92
C67
C62
C50
C34
C27
C26
C28
C33
C85
C63
C89
C76
C51
C56
C95
C84
C91
C64
C66
C83
C57
C58
C75
C77
C25
C18
C48
C49
C19
C65
C52
C86
C74
C59
C87
C97
C13
C9
C99
C82
C90
C53
C69
C98
C78
C79
C73
C60
C88
C55
C7
C6
C8
C46
C4
C45
C43
C44
C42
C40
C36
C35
C37
C39
C38
C41
C17
C47
C14
C12
C16
C15
C11
C10

rc102_21


S0
S1
S2
S3
S4
S5
S6
S7
S8
S9
S10
S11
S12
S13
S14
S15
S16
S17
S18
S19
S20
C1
C2
C3
C4
C5
C6
C7
C8
C9
C10
C11
C12
C13
C14
C15
C16
C17
C18
C19
C20
C21
C22
C23
C24
C25
C26
C27
C28
C29
C30
C31
C32
C33
C34
C35
C36
C37
C38
C39
C40
C41
C42
C43
C44
C45
C46
C47
C48
C49
C50
C51
C52
C53
C54
C55
C56
C57
C58
C59
C60
C61
C62
C63
C64
C65
C66
C67
C68
C69
C70
C71
C72
C73
C74
C75
C76
C77
C78
C79
C80
C81
C82
C83
C84
C85
C86
C87
C88
C89
C90
C91
C92
C93
C94
C95
C96
C97
C98
C99
C100