Tutorial for VehicleRoutingSynPre

Payakorn Saksuriya

April 29, 2022

Import package

```
using VehicleRoutingSynPre
using Latexify
```

---

1 Benchmark

Benchmark name is in the form, for exmaple, ”ins10-1.jld2” is the instance number 1 with 10 nodes (not include depot).

To load the instance data use

```
name = "ins10-1"
num_node, num_vehi, num_serv, mind, maxd, a, r, d, p, e, l = load_data(name);
```

where

- num_node: total number of nodes (including depot)
- num_vehi: total number of vehicles
- num_serv: total number of services
- mind: minumum different of starting time between two services of each node
- maxd: maximum different of starting time between two services of each node
- a: compatibility matrix a[i,j] = 1 if vehicle i can process service j
- r: requiment matrix r[i,j] = 1 if node i requires service j
- d: distance matrix
- p: processing time matrix, p[i, j, k] = processing time of service j of vehicle i on node k
- e: earilest start time of each node
- l: latest start time of each node

Generate random particle (solution)

```
particle = generate_particles("ins10-1");
```

```
11, [6]
7, [5]
4, [2]
6, [3]
3, [5]
10, [1]
2, [4]
5, [4]
8, [3]
```

The particle is `struct` of `Particle` with fields

```
dump(Particle)
```

```
Particle <: Any
  route::Vector{Vector{Vector{Int64}}}
  starttime::Dict{Int64, Array{Float64}}
  slot::Dict{Int64, Vector{Int64}}
  serv_a::Tuple
  serv_r::Dict{Int64, Vector{Int64}}
  num_node::Int64
  num_vehi::Int64
  num_serv::Int64
  mind::Vector{Float64}
  maxd::Vector{Float64}
  a::Array{Int64}
  r::Array{Int64}
  d::Matrix{Float64}
  p::Array{Float64}
  e::Vector{Int64}
  l::Vector{Int64}
  PRE::Vector{Tuple}
  SYN::Vector{Tuple}
```

```
particle.route
```

```
3-element Vector{Vector{Vector{Int64}}}:
 [[4, 2], [6, 3], [10, 1], [8, 3], [1, 1]]
 [[9, 6], [11, 6], [7, 5], [1, 1]]
 [[9, 5], [3, 5], [2, 4], [5, 4], [1, 1]]
```

```
particle.slot
```

```
Dict{Int64, Vector{Int64}} with 10 entries:
  5  => [4]
  4  => [2]
  6  => [3]
  7  => [5]
  2  => [4]
  10 => [1]
  11 => [6]
  9  => [5, 6]
  8  => [3]
  3  => [5]
```

```
print(latexify("x/y"))
```

```
$\frac{x}{y}$
```

Total distance

```
total_distance(particle)
```

```
560.1594410000001
```