Artificial Intelligence

Narendiran S

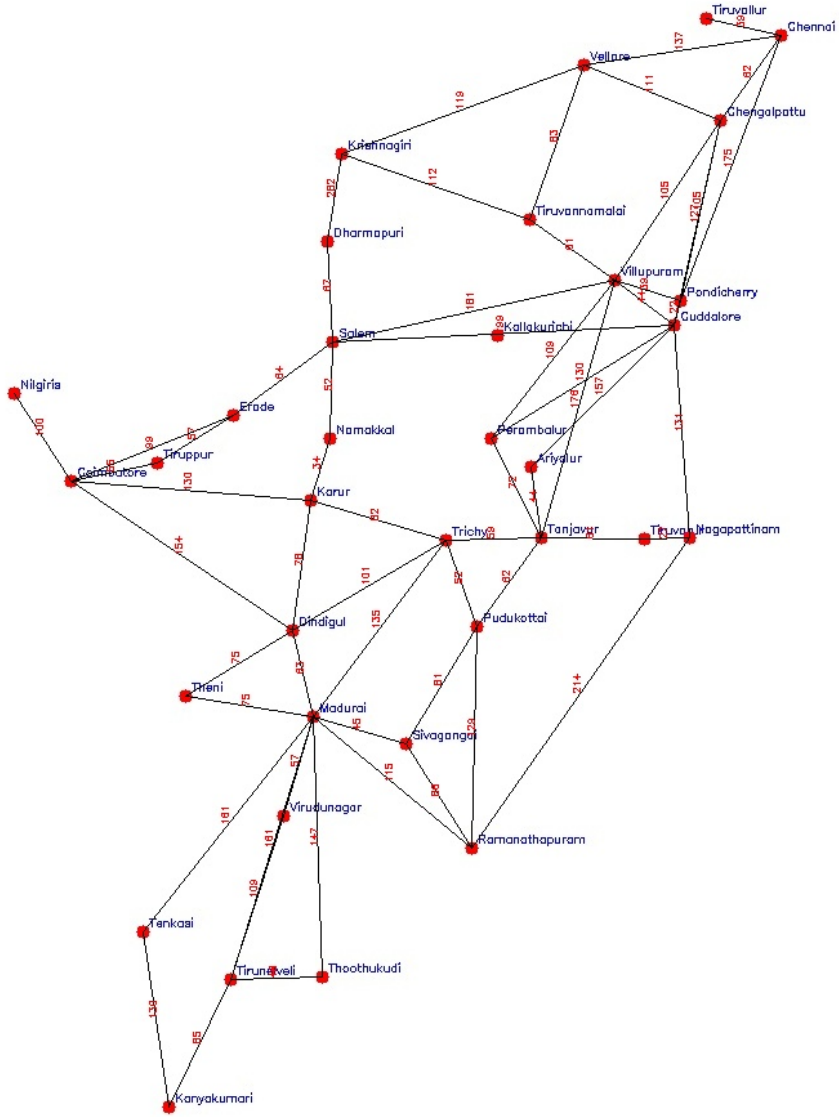
Department of Electronics and Communication Engineering
National Institute of Technology, Calicut

August 11, 2020

Assumptions

- Gray color represents Unvisited Node.
- Blue color represents Notes in Frontier.
- Orange color represents Visited Nodes.
- Black color represents Failure Nodes.
- Green color represents Success Node.

The map I choose is Tamil Nadu. The path is from *Coimbatore* to *Pondicherry*.

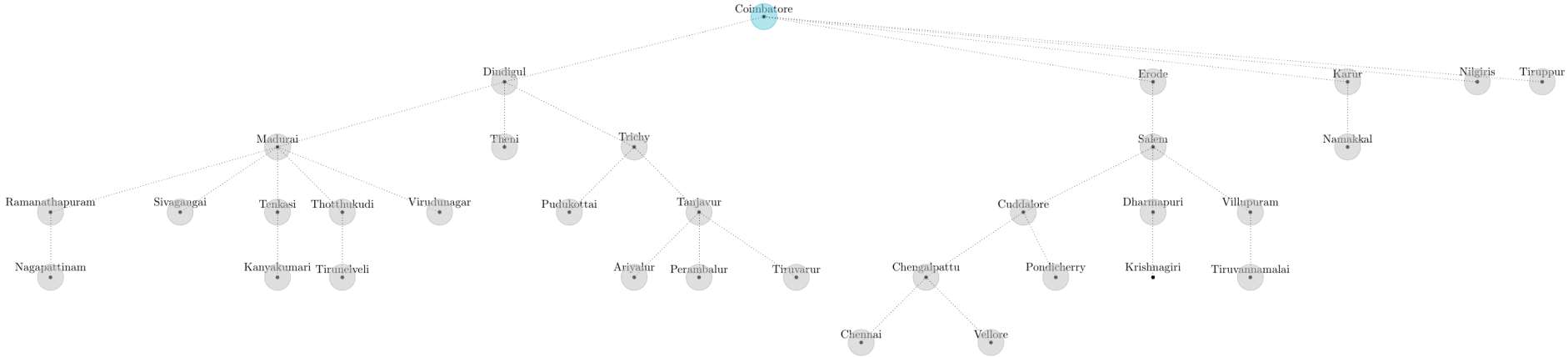


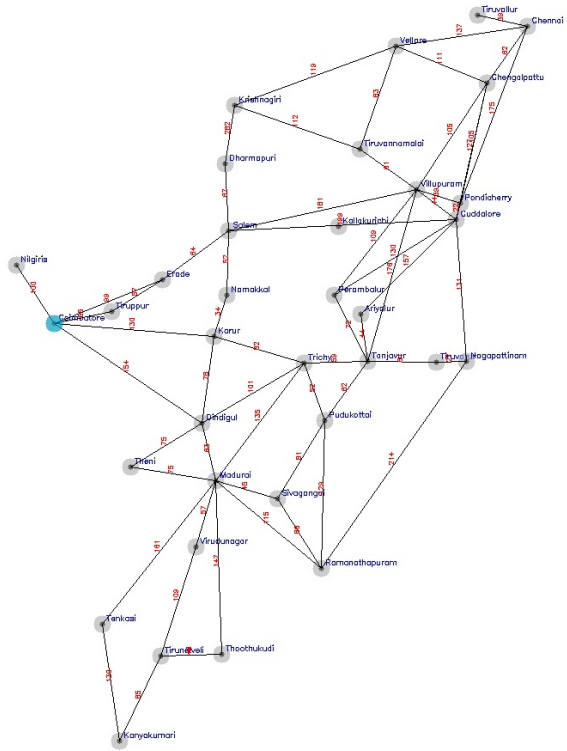
Breath First Search - BFS

- In BFS, we expand the shallowest node first.
- It doesn't use any domain information.
- The data structure used is a **QUEUE**.

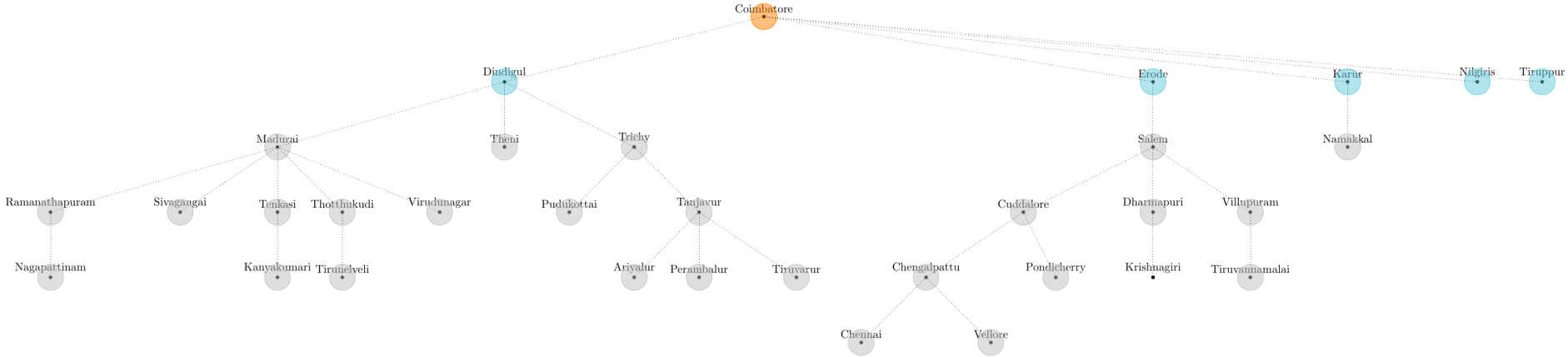
```
function BREADTH-FIRST-SEARCH(initialState, goalTest)
  returns SUCCESS or FAILURE:

  frontier = Queue.new(initialState)
  explored = Set.new()

  while not frontier.isEmpty()
    state = frontier.dequeue()
    explored.add(state)
    if goalTest(state):
      return SUCCESS(state)

    for neighbor in state.neighbors():
      if neighbor not in frontier U explored:
        frontier.enqueue(neighbor)
  return FAILURE
```

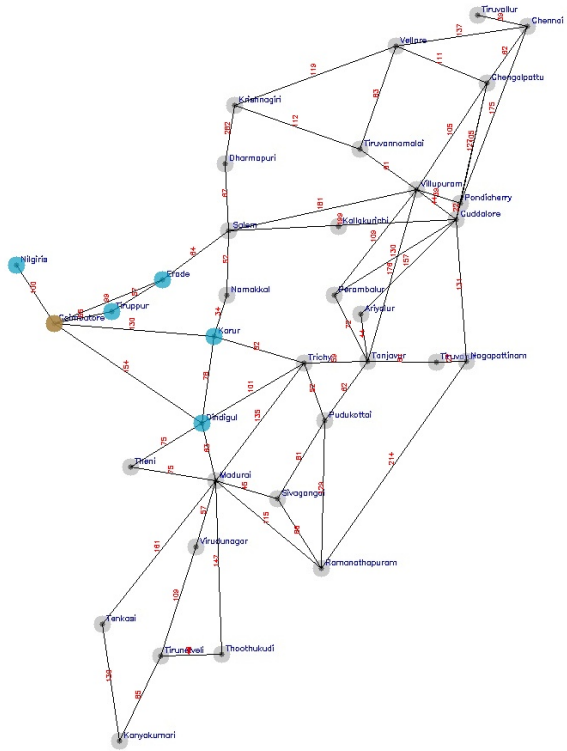
Fontier: ['Coimbatore']



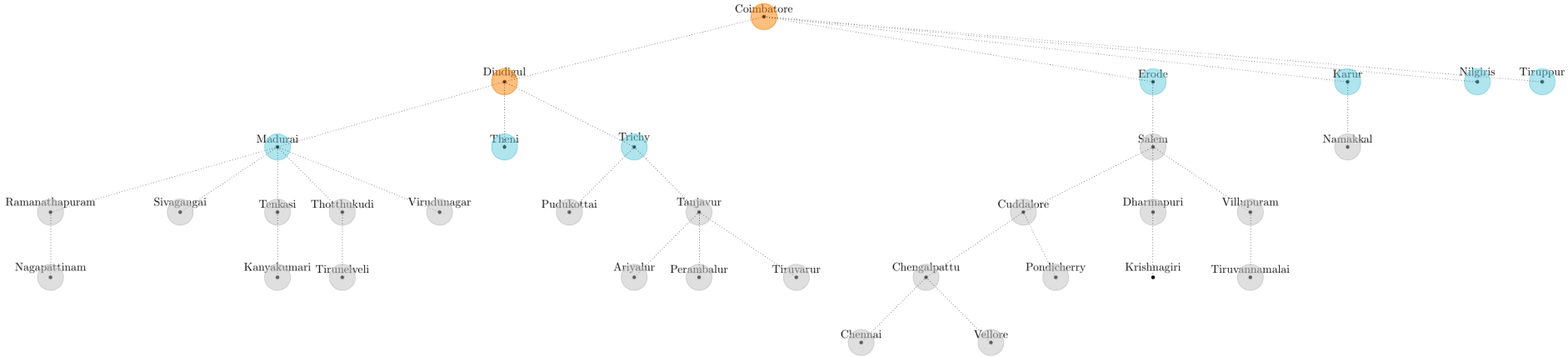


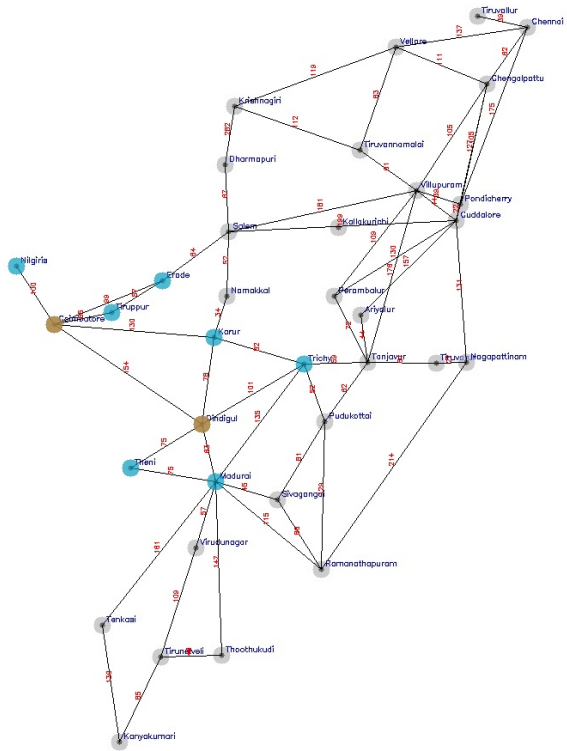
Fontier: ['Dindigul', 'Erode', 'Karur', 'Nilgiris', 'Tiruppur']



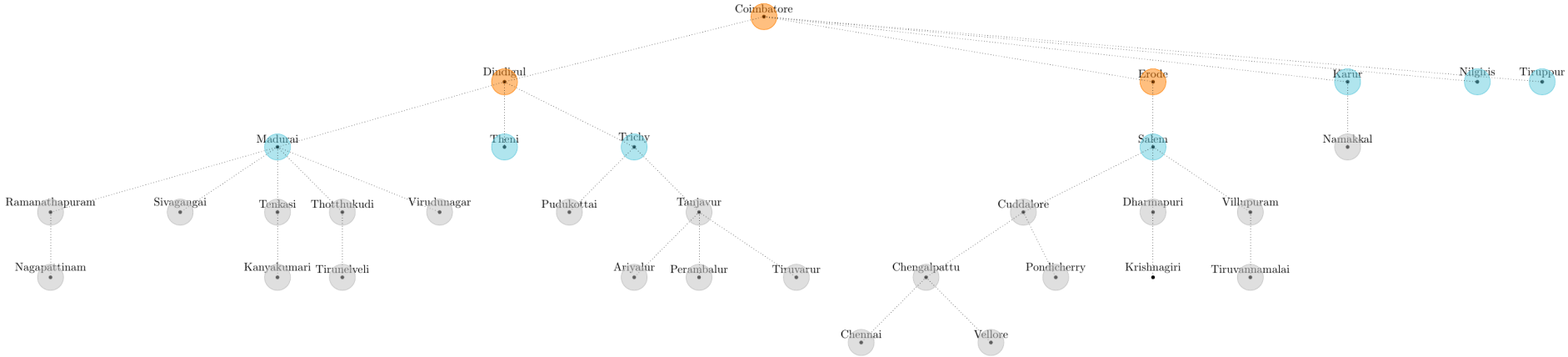


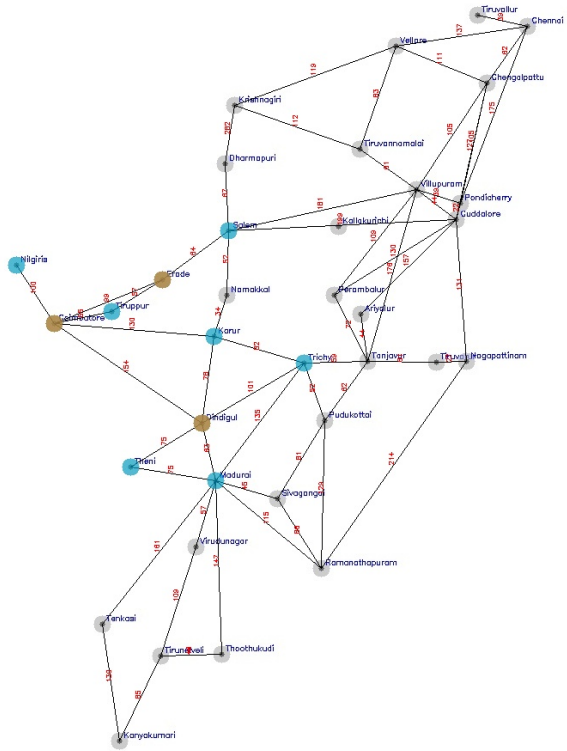
Fontier: ['Erode', 'Karur', 'Nilgiris', 'Tiruppur', 'Madurai', 'Theni', 'Trichy']



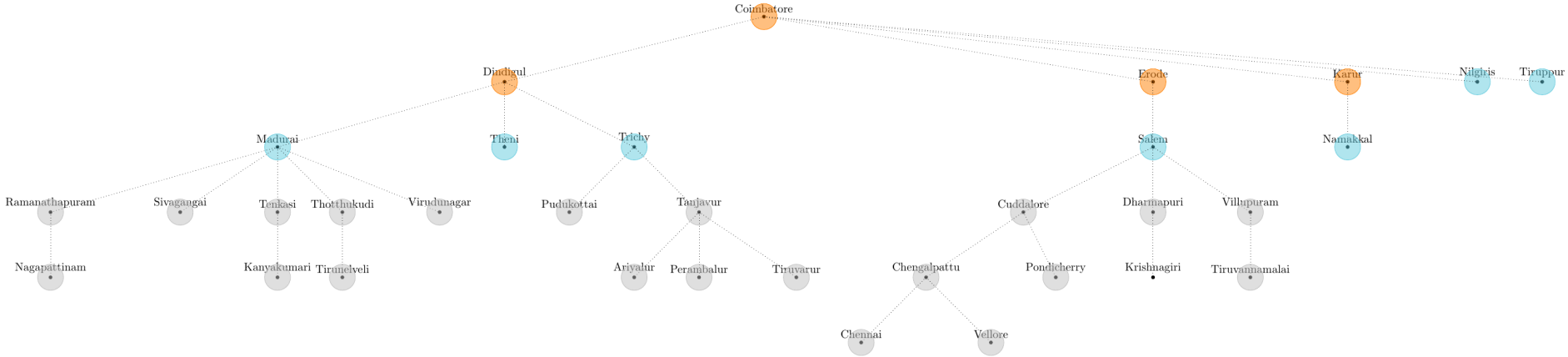


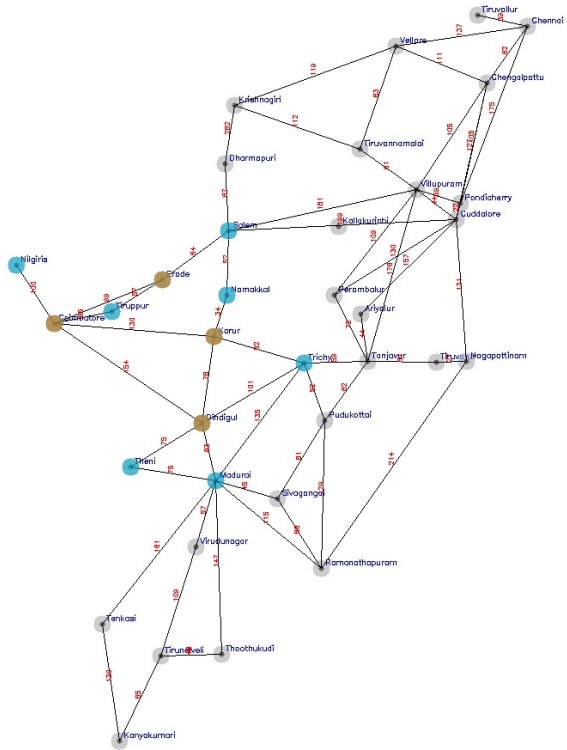
Fontier: ['Karur', 'Nilgiris', 'Tiruppur', 'Madurai', 'Theni', 'Trichy', 'Salem']



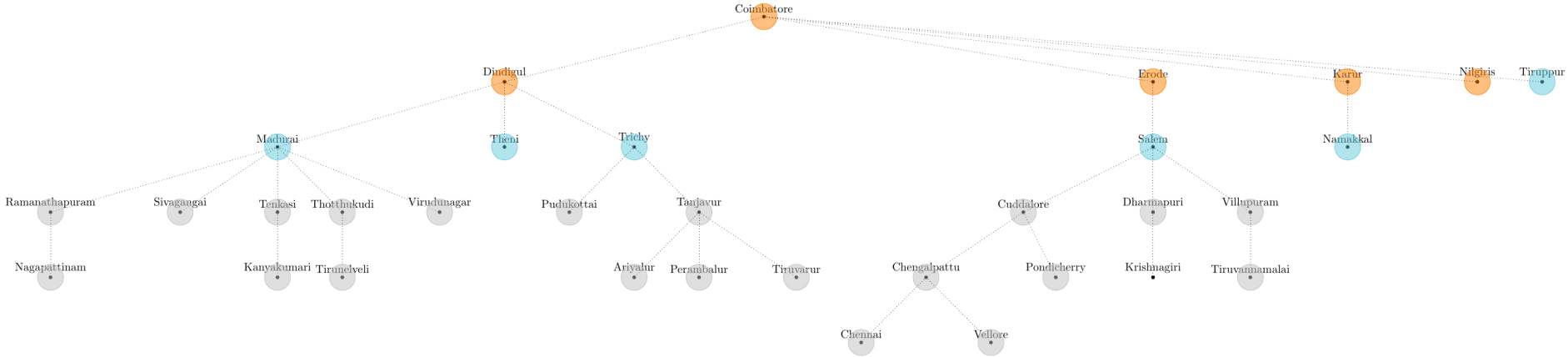


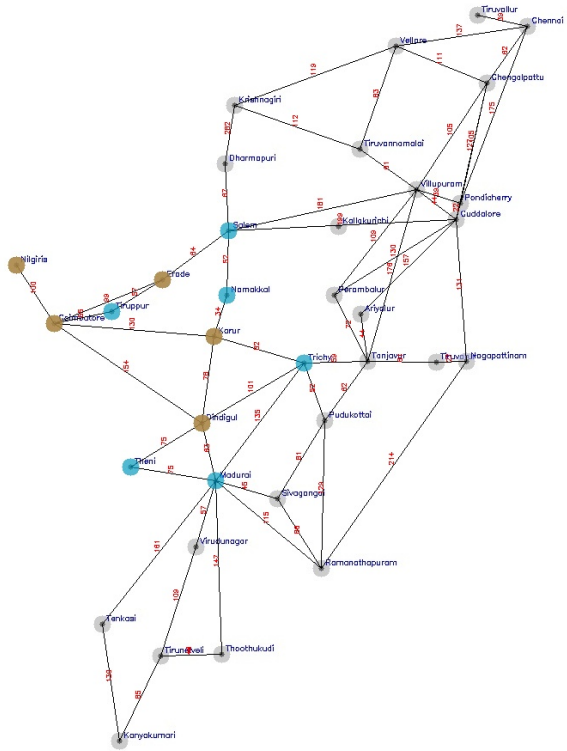
Fontier: ['Nilgiris', 'Tiruppur', 'Madurai', 'Theni', 'Trichy', 'Salem', 'Namakkal']



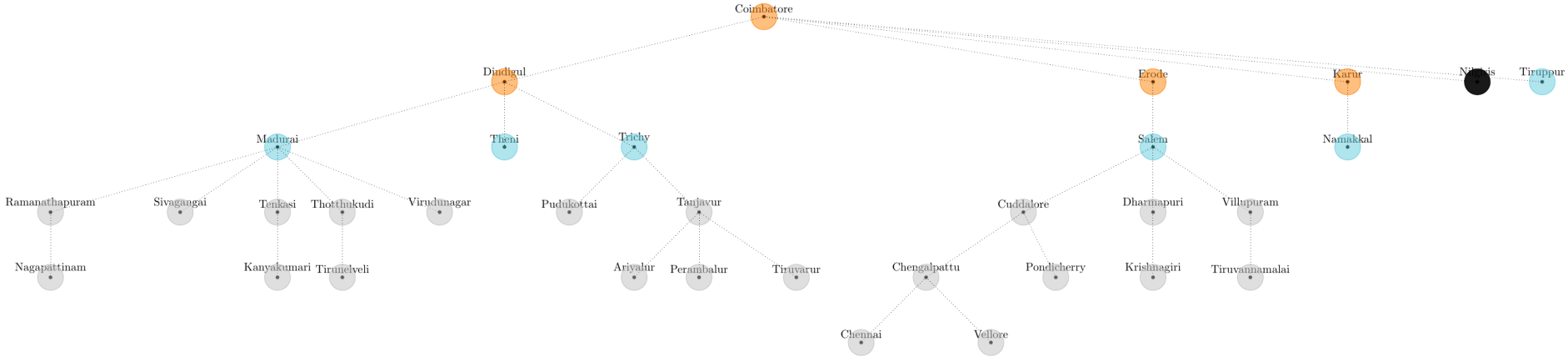


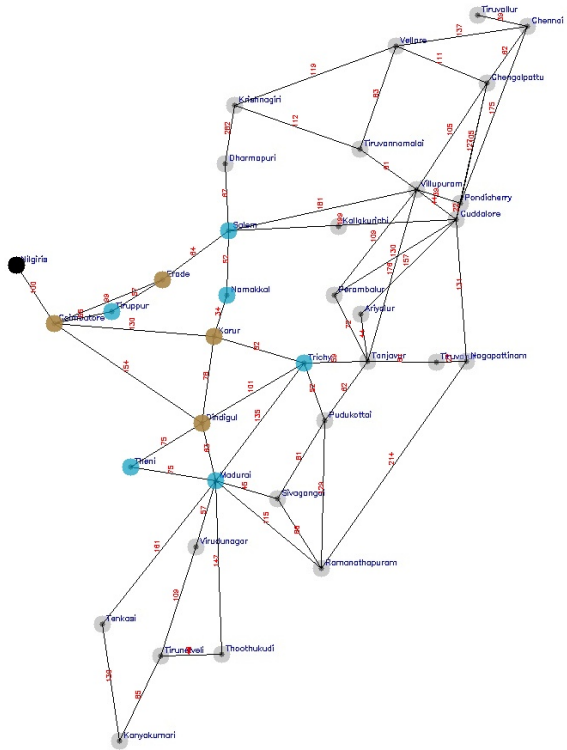
Fontier: ['Tiruppur', 'Madurai', 'Theni', 'Trichy', 'Salem', 'Namakkal']



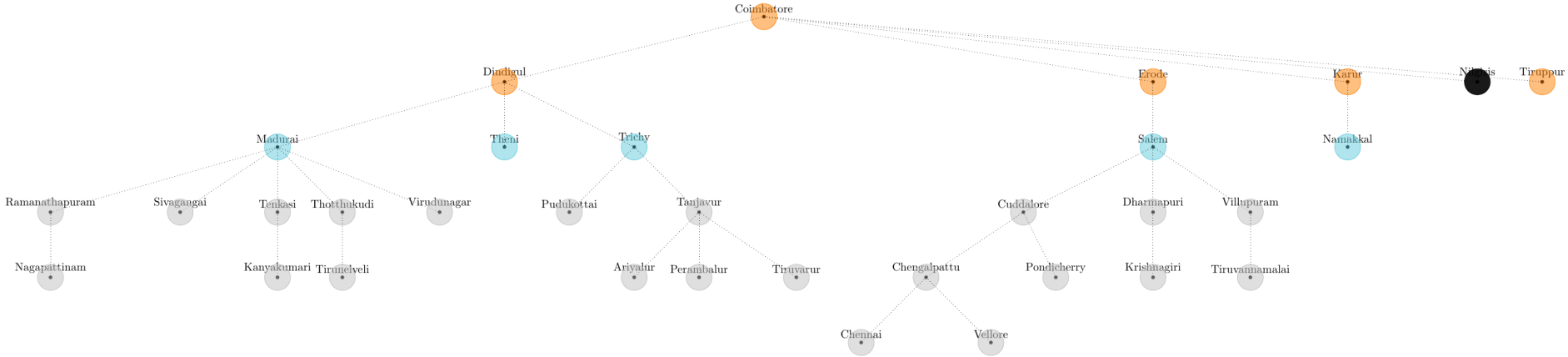


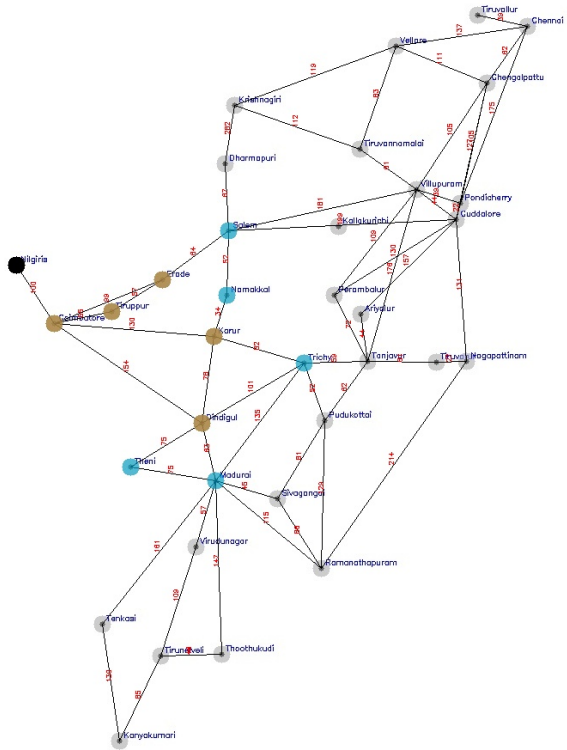
Fontier: ['Tiruppur', 'Madurai', 'Theni', 'Trichy', 'Salem', 'Namakkal']



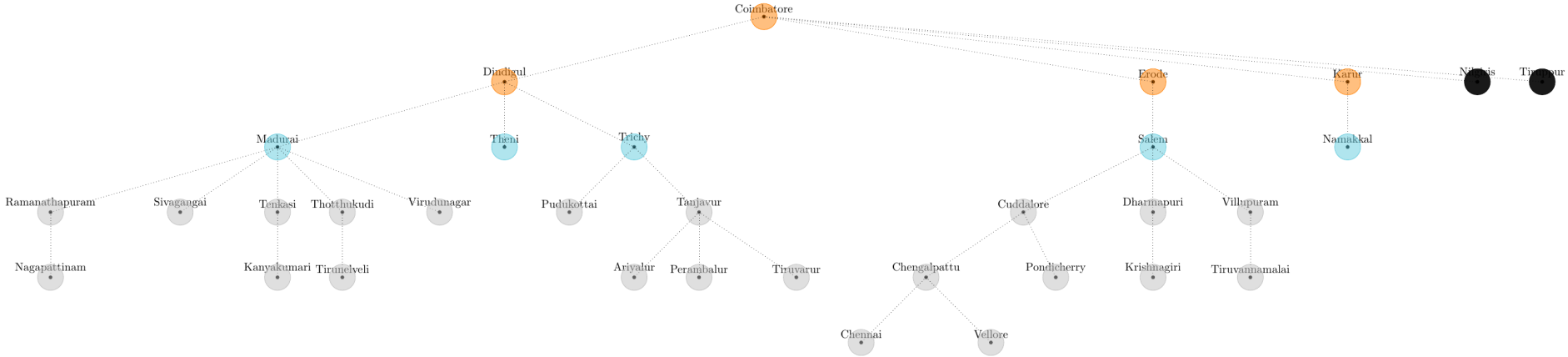


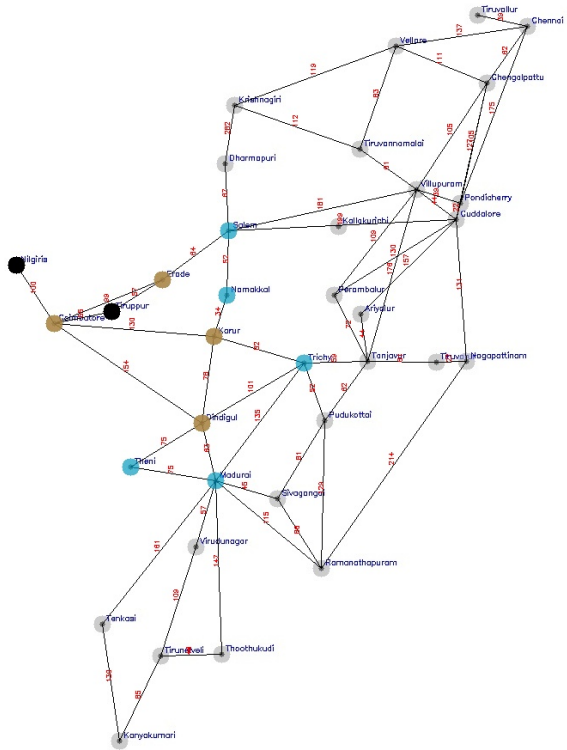
Fontier: ['Madurai', 'Theni', 'Trichy', 'Salem', 'Namakkal']



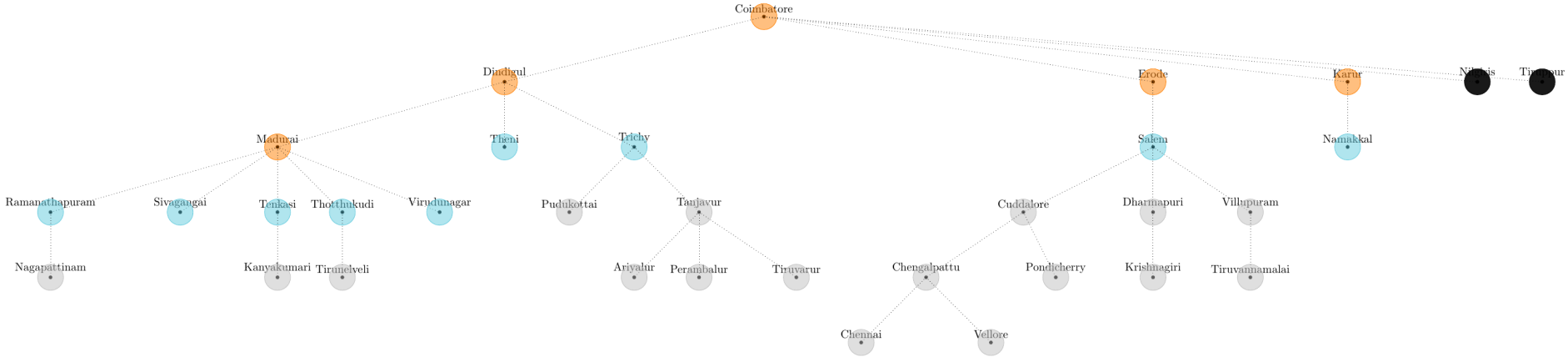


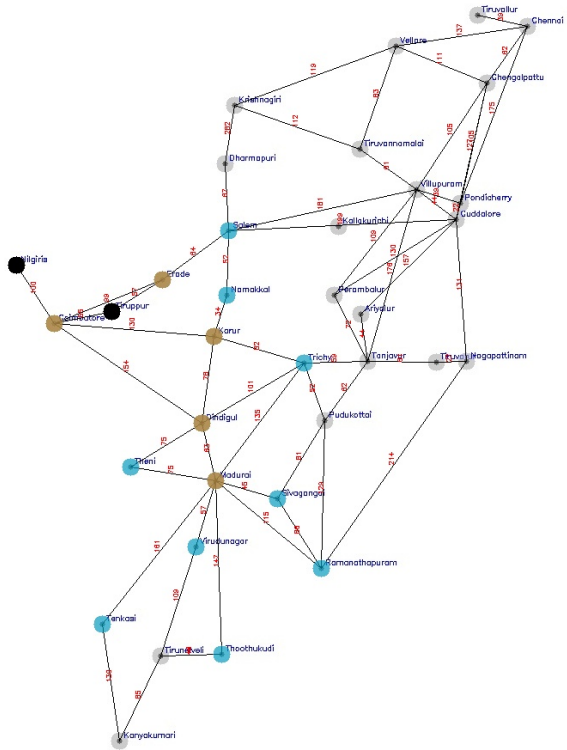
Fontier: ['Madurai', 'Theni', 'Trichy', 'Salem', 'Namakkal']



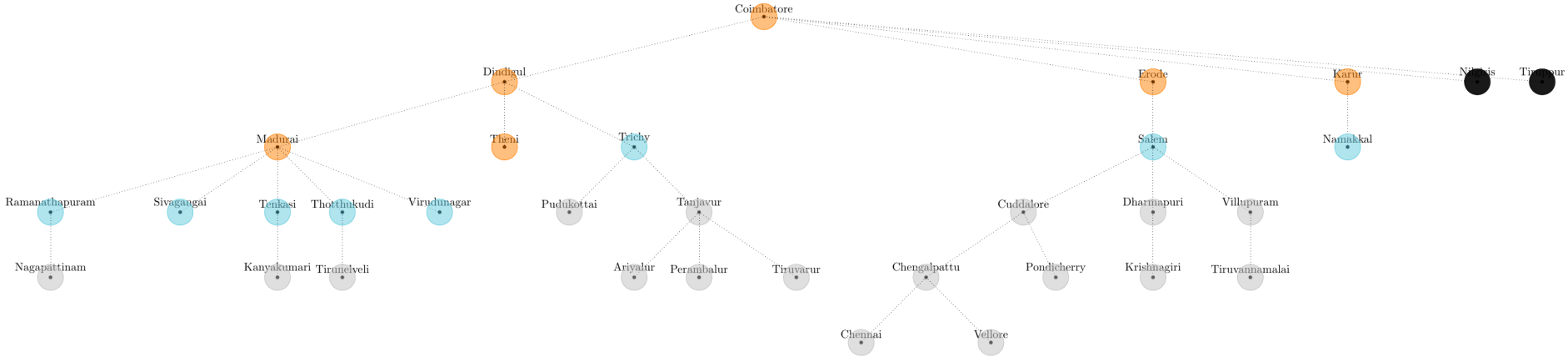


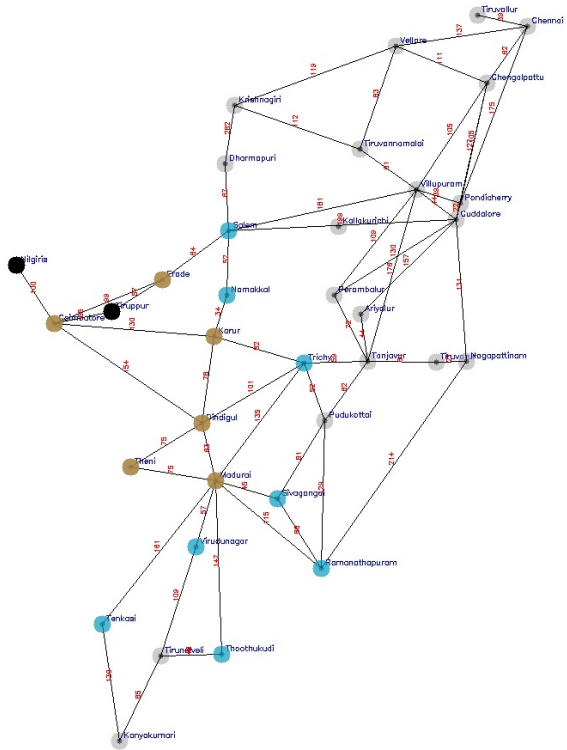
Fontier: ['Theni', 'Trichy', 'Salem', 'Namakkal', 'Ramanathapuram', 'Sivagangai', 'Tenkasi', 'Thoothukudi', 'Virudunagar']



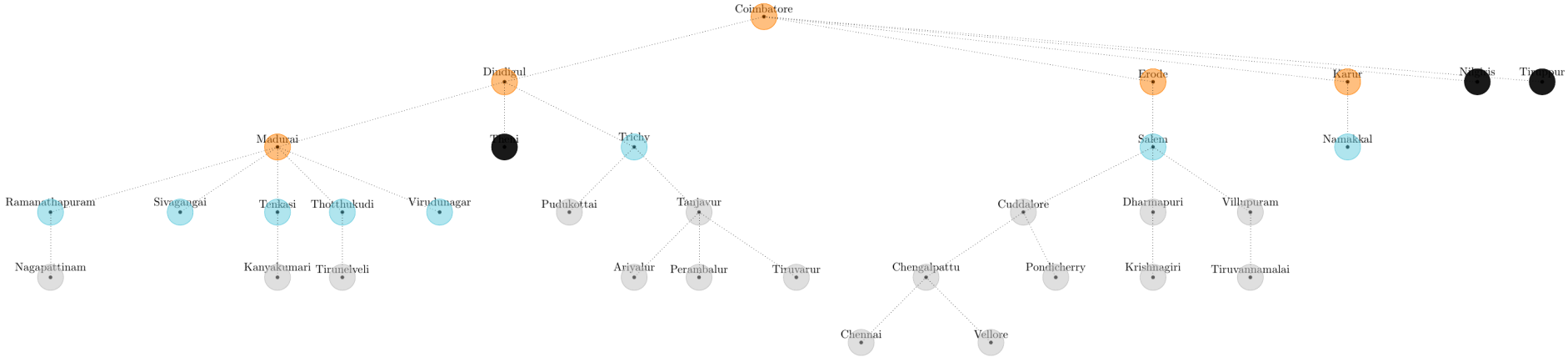


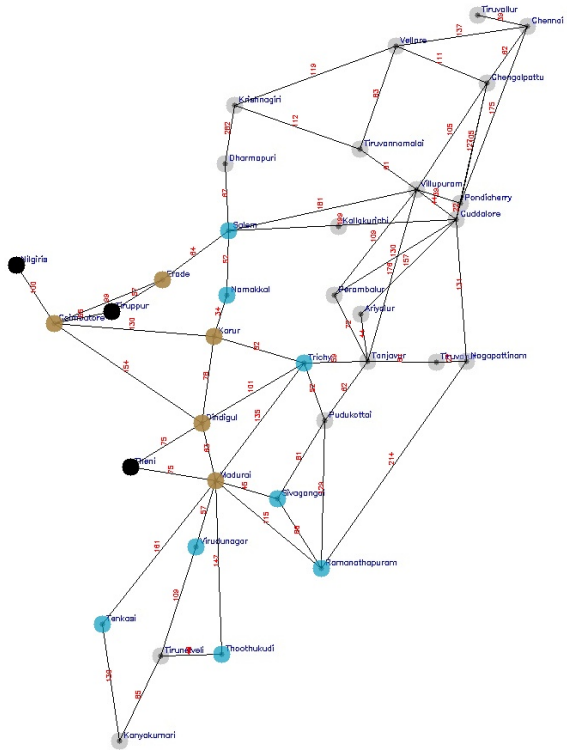
Fontier: ['Trichy', 'Salem', 'Namakkal', 'Ramanathapuram', 'Sivagangai', 'Tenkasi', 'Thoothukudi', 'Virudunagar']



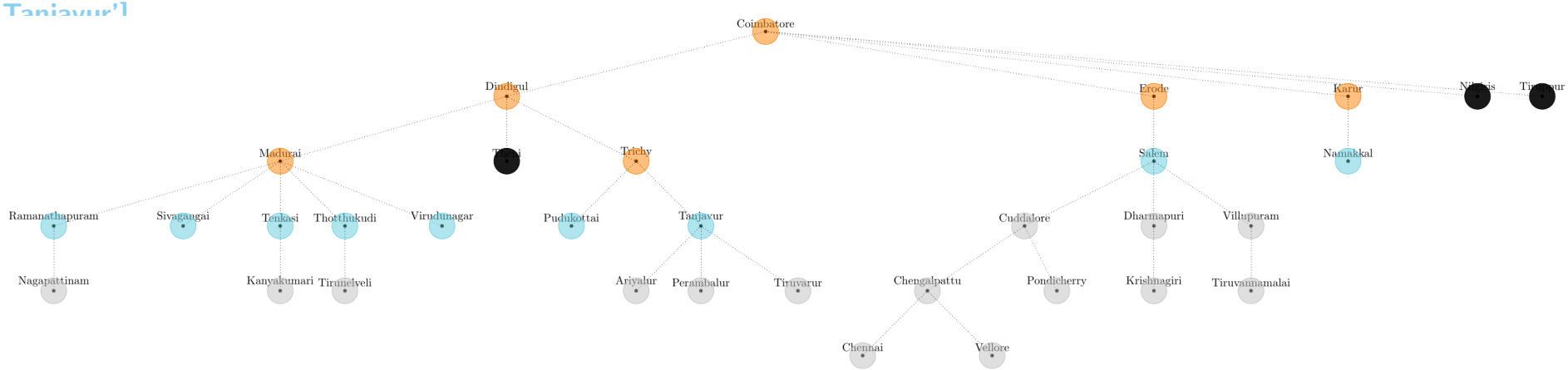


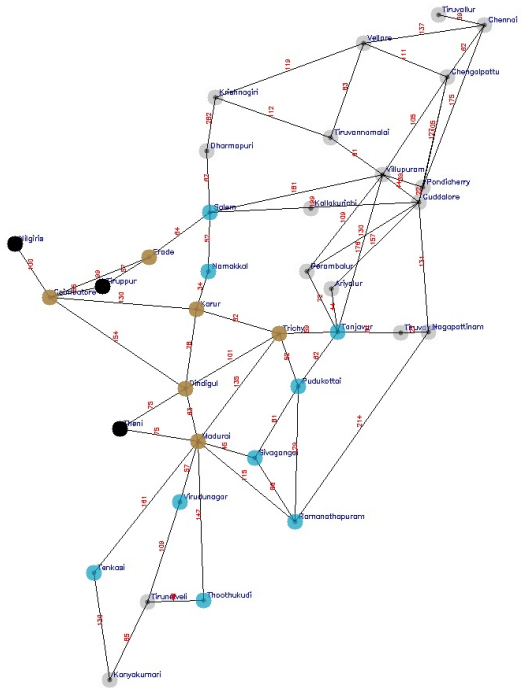
Fontier: ['Trichy', 'Salem', 'Namakkal', 'Ramanathapuram', 'Sivagangai', 'Tenkasi', 'Thoothukudi', 'Virudunagar']



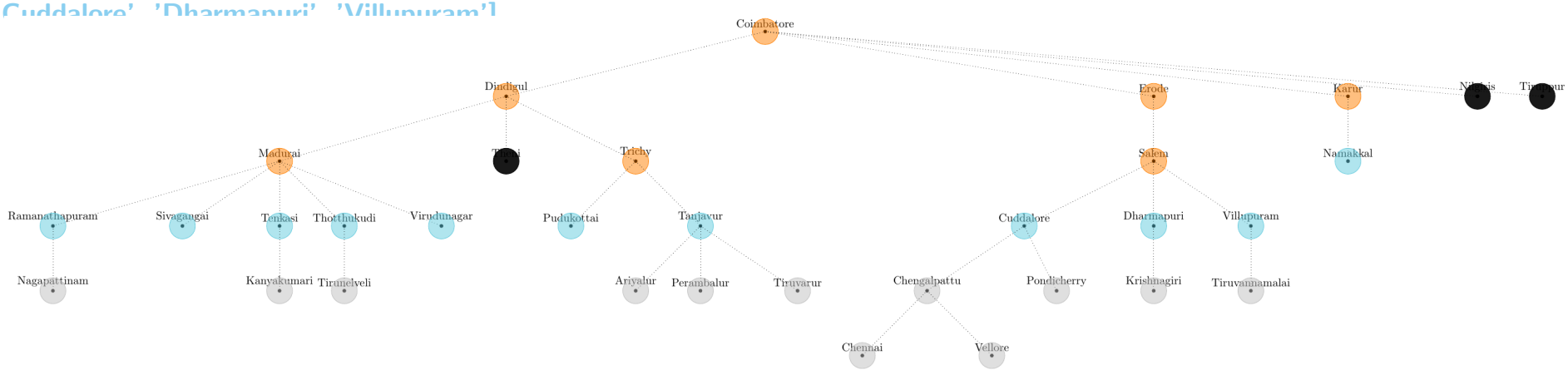


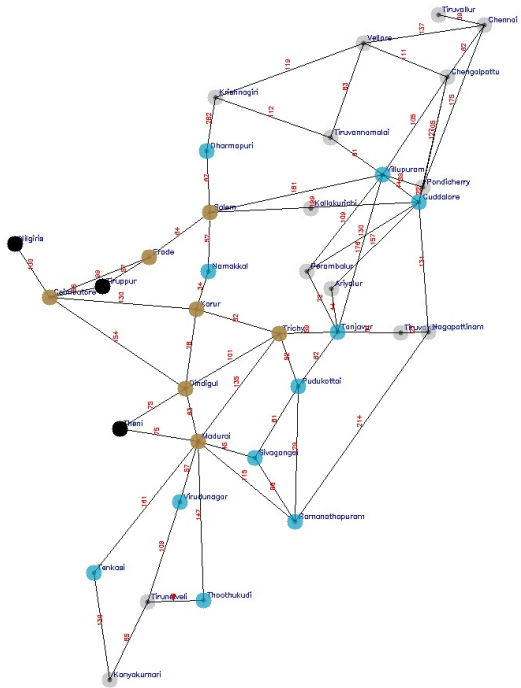
Fontier: ['Salem', 'Namakkal', 'Ramanathapuram', 'Sivagangai', 'Tenkasi', 'Thoothukudi', 'Virudunagar', 'Pudukottai', 'Tanjavur']



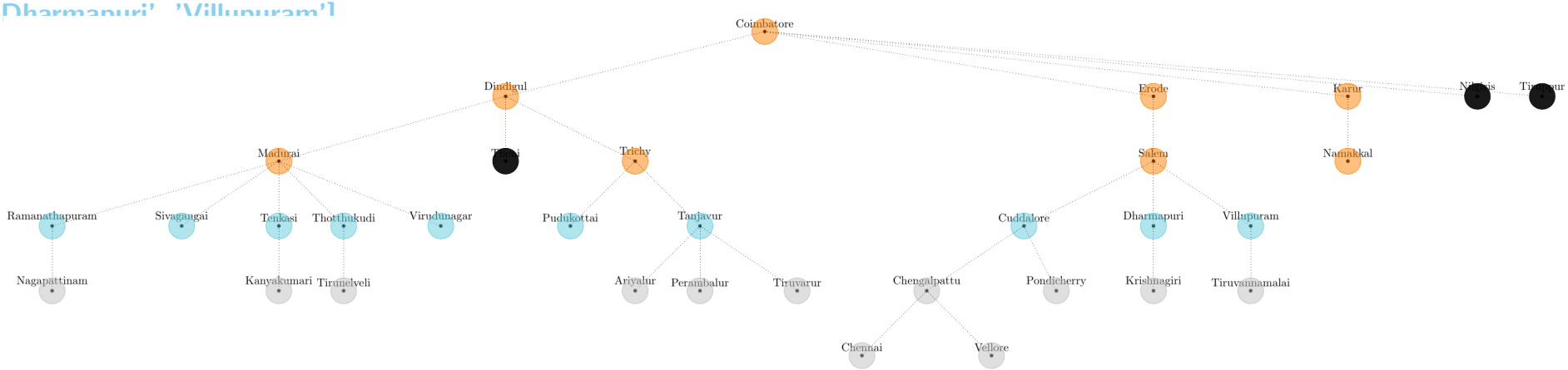


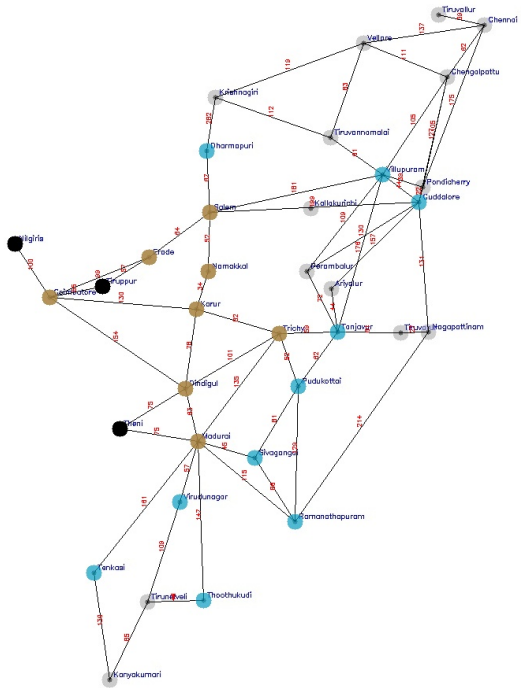
Fontier: ['Namakkal', 'Ramanathapuram', 'Sivagangai', 'Tenkasi', 'Thoothukudi', 'Virudunagar', 'Pudukottai', 'Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram']



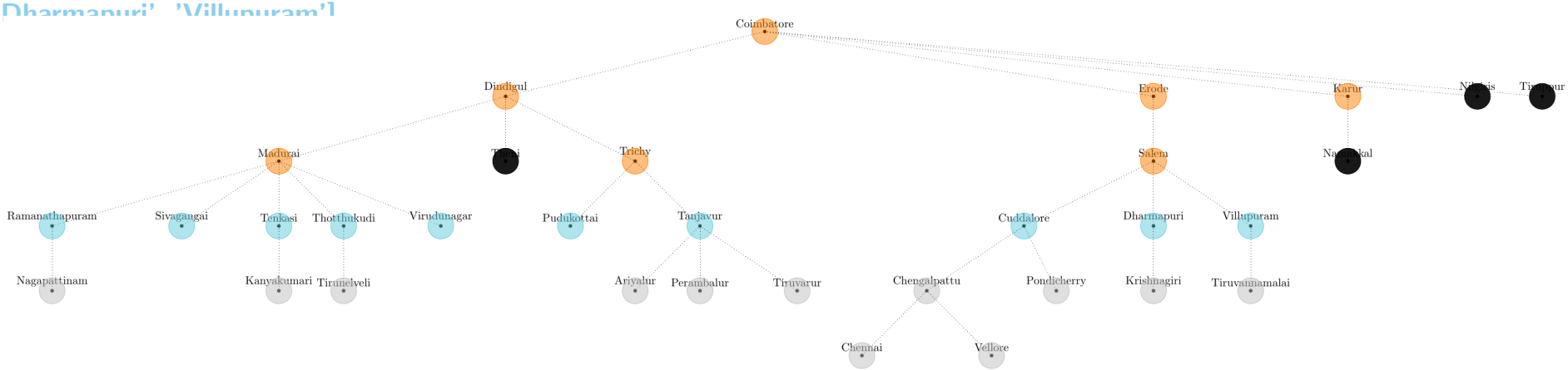


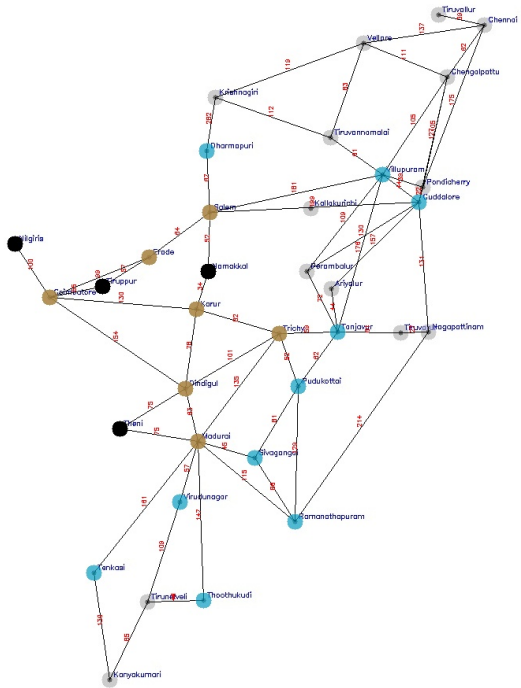
Fontier: ['Ramanathapuram', 'Sivagangai', 'Tenkasi', 'Thoothukudi', 'Virudunagar', 'Pudukottai', 'Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram']



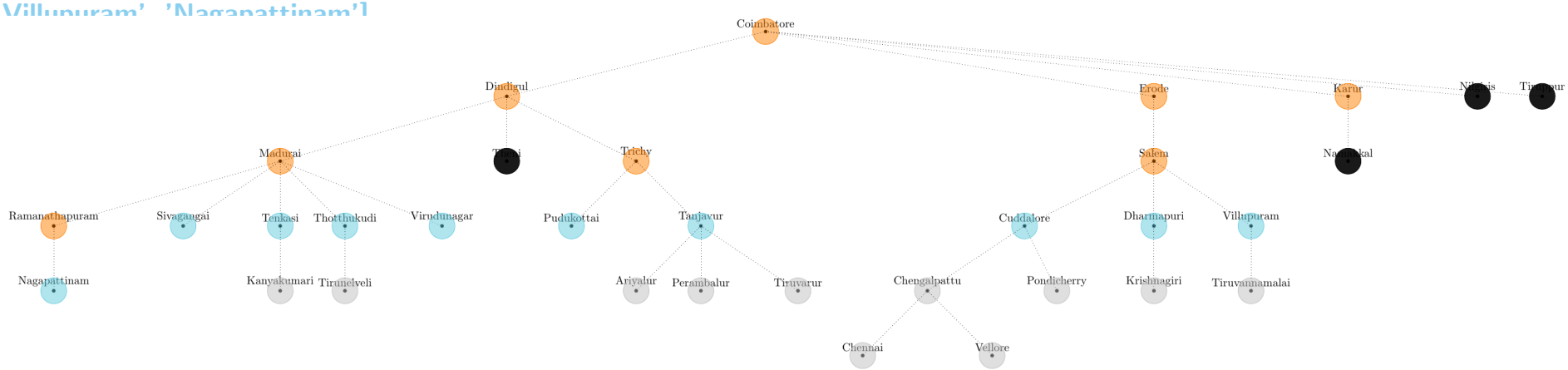


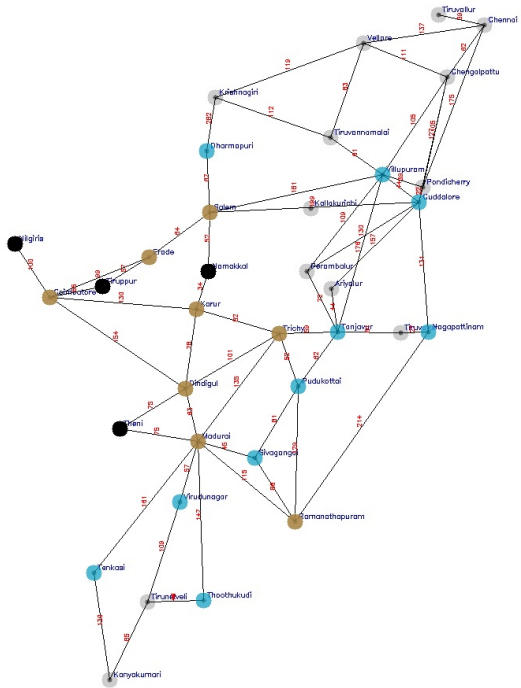
Fontier: ['Ramanathapuram', 'Sivagangai', 'Tenkasi', 'Thoothukudi', 'Virudunagar', 'Pudukottai', 'Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram']



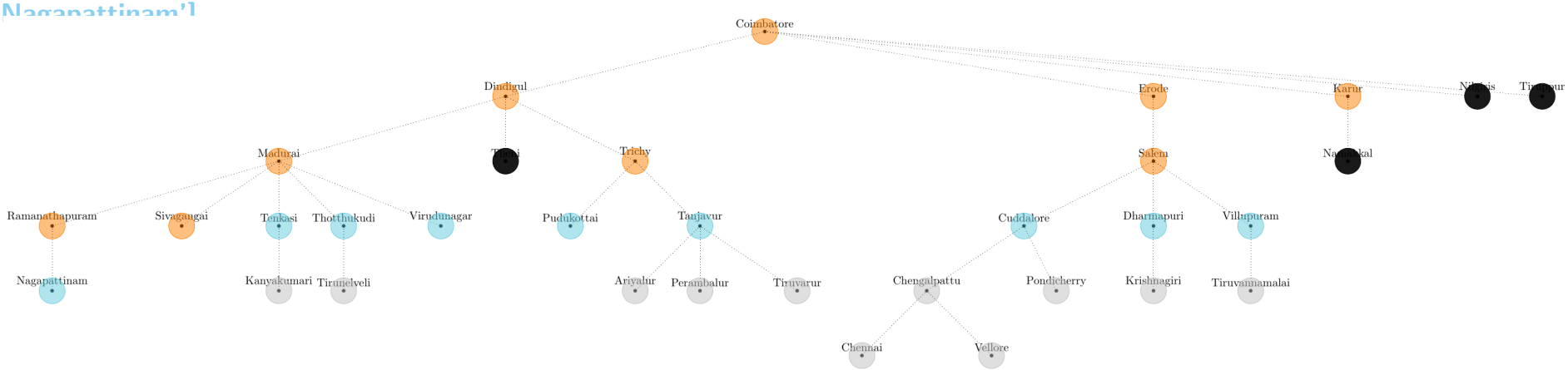


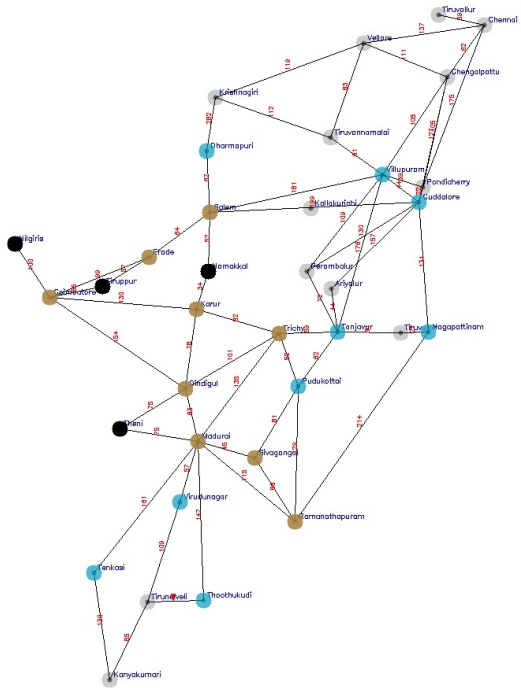
Fontier: ['Sivagangai', 'Tenkasi', 'Thoothukudi', 'Virudunagar', 'Pudukottai', 'Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram', 'Nagapattinam']



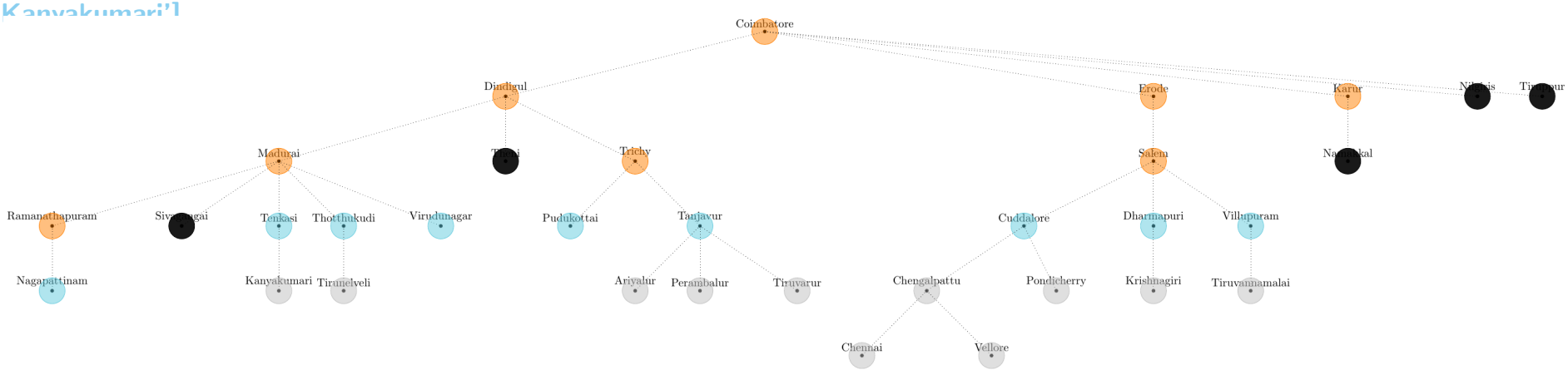


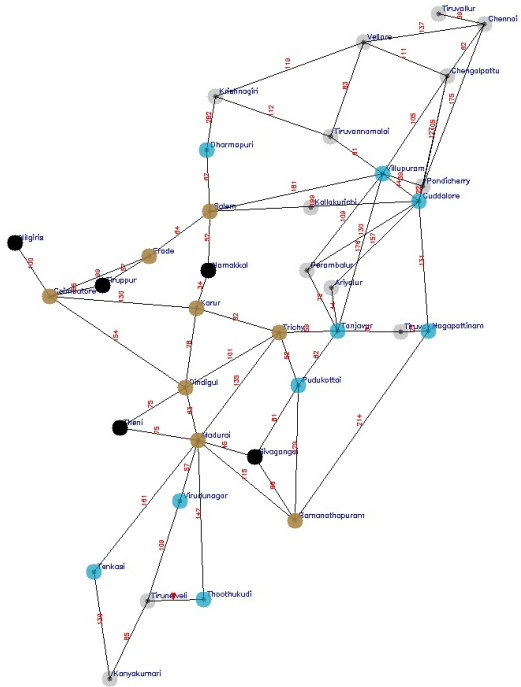
Fontier: ['Tenkasi', 'Thoothukudi', 'Virudunagar', 'Pudukottai', 'Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram', 'Nagapattinam']



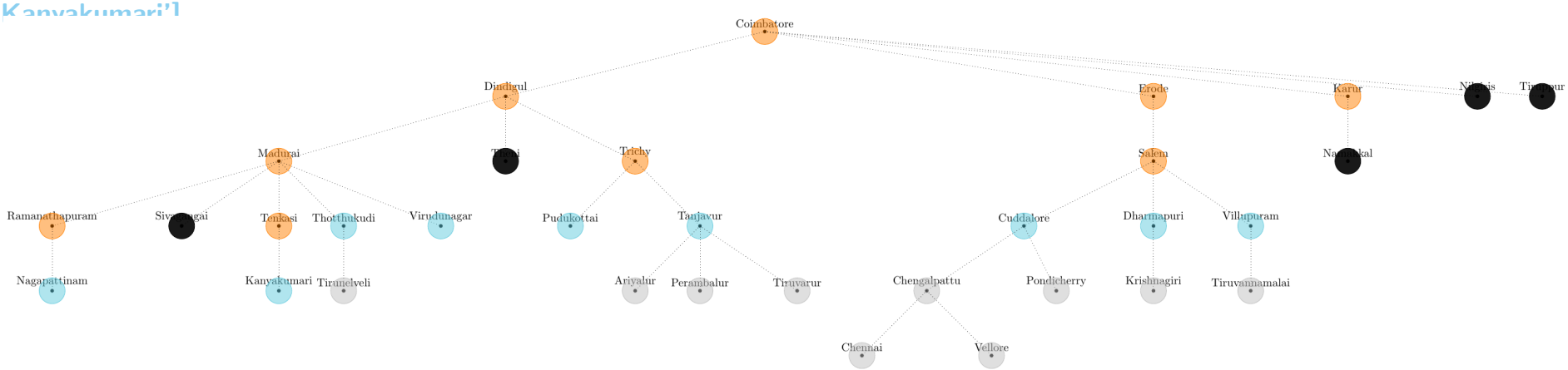


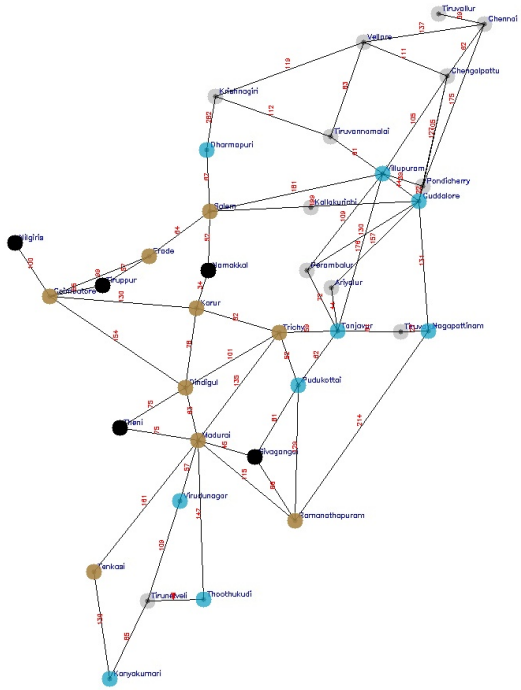
Fontier: ['Thoothukudi', 'Virudunagar', 'Pudukottai', 'Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram', 'Nagapattinam', 'Kanyakumari']



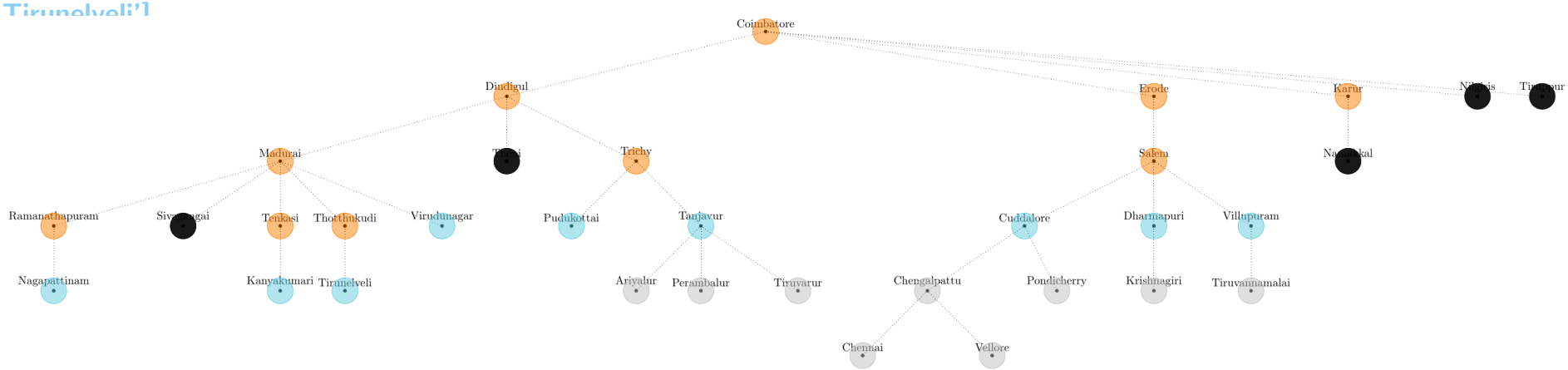


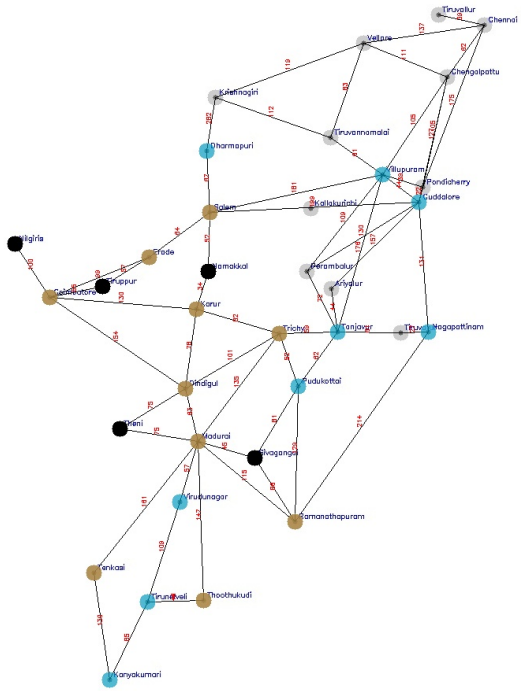
Fontier: ['Thoothukudi', 'Virudunagar', 'Pudukottai', 'Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram', 'Nagapattinam', 'Kanyakumari']



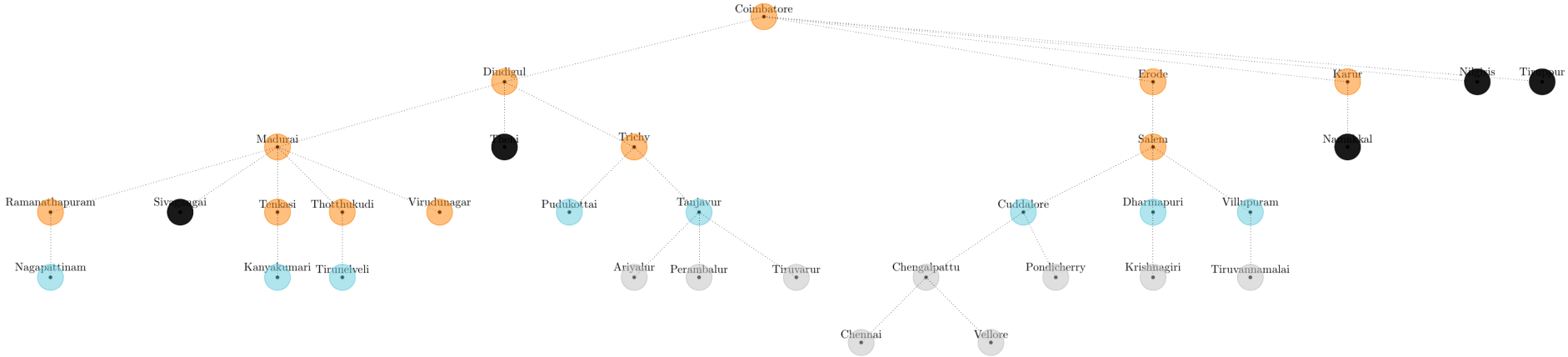


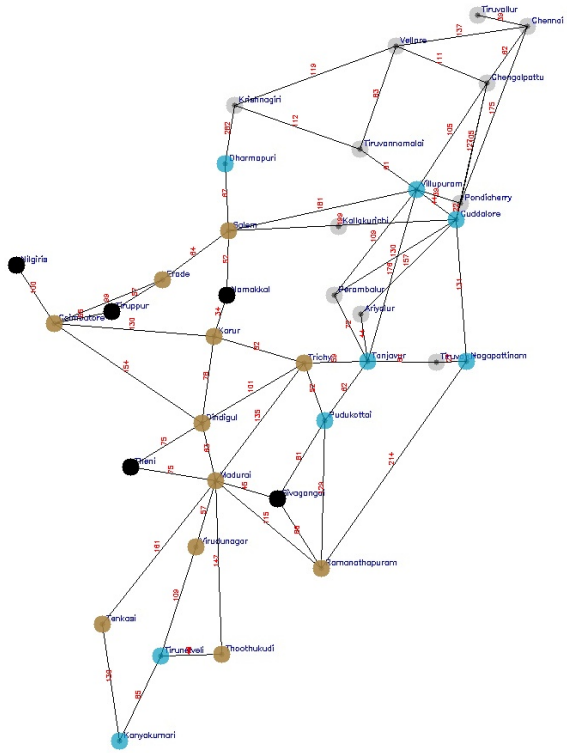
Fontier: ['Virudunagar', 'Pudukottai', 'Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram', 'Nagapattinam', 'Kanyakumari', 'Tirunelveli']



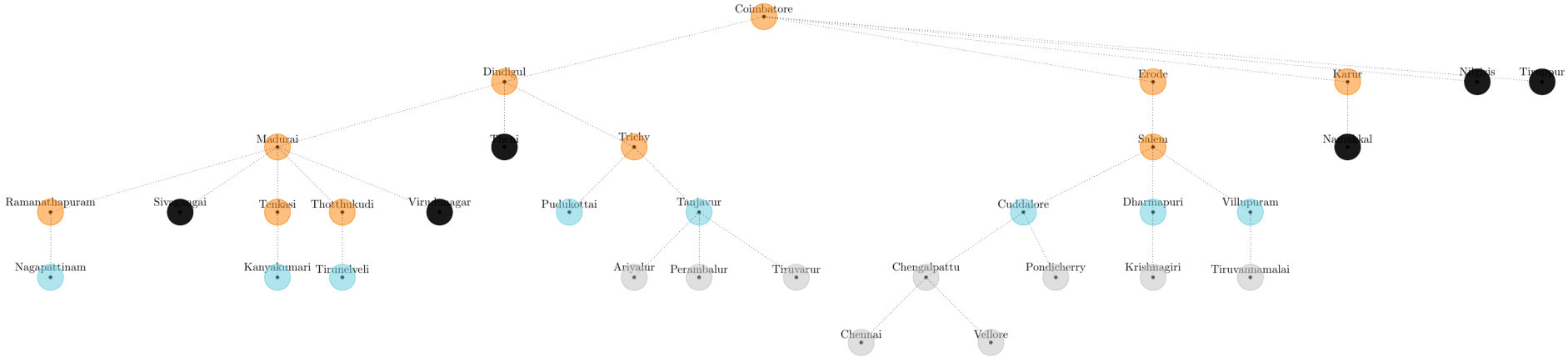


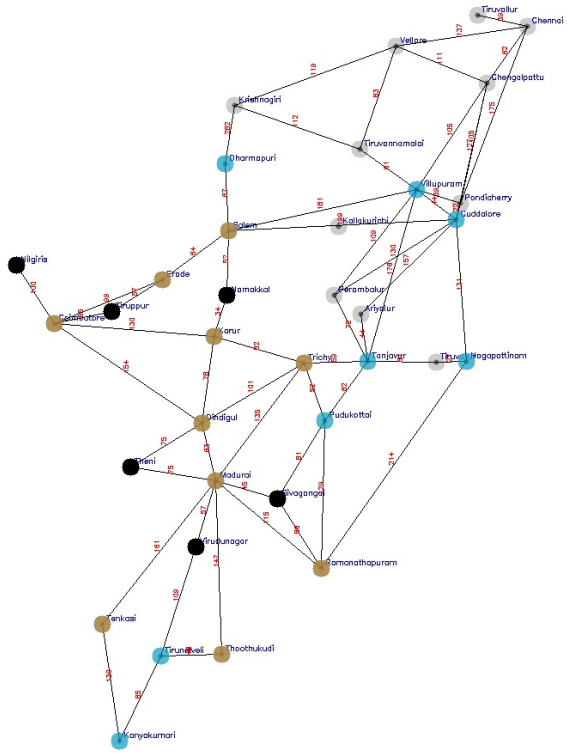
Fontier: ['Pudukottai', 'Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram', 'Nagapattinam', 'Kanyakumari', 'Tirunelveli']



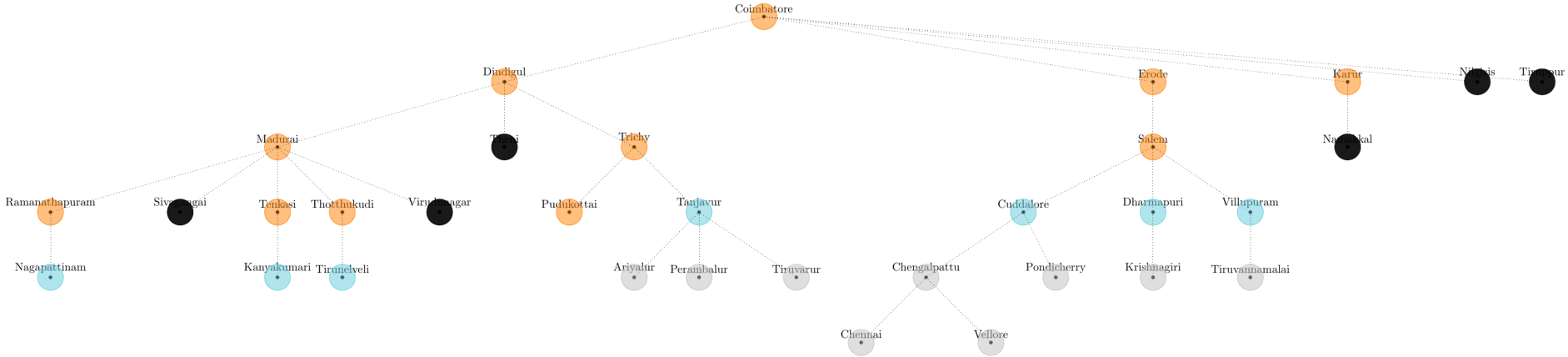


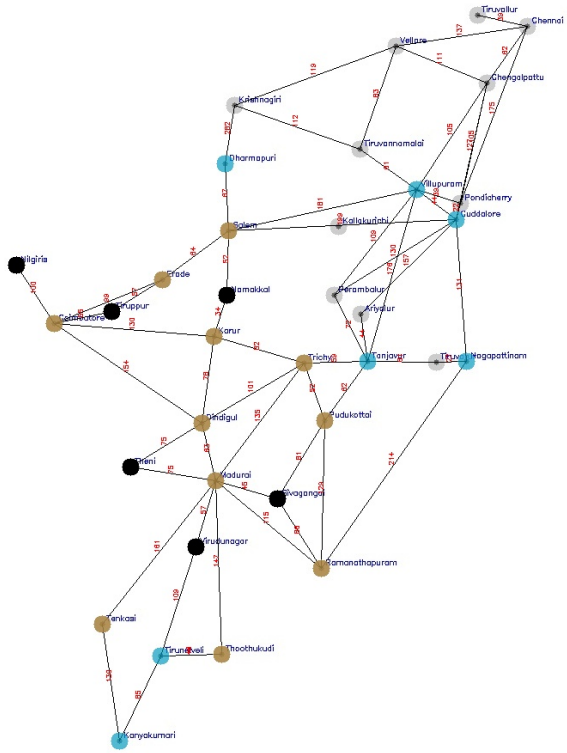
Fontier: ['Pudukottai', 'Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram', 'Nagapattinam', 'Kanyakumari', 'Tirunelveli']



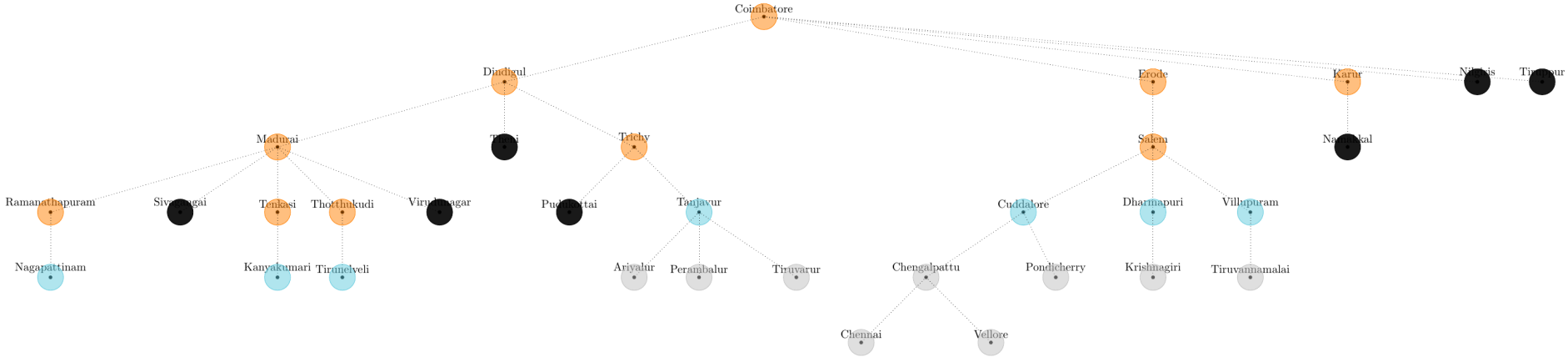


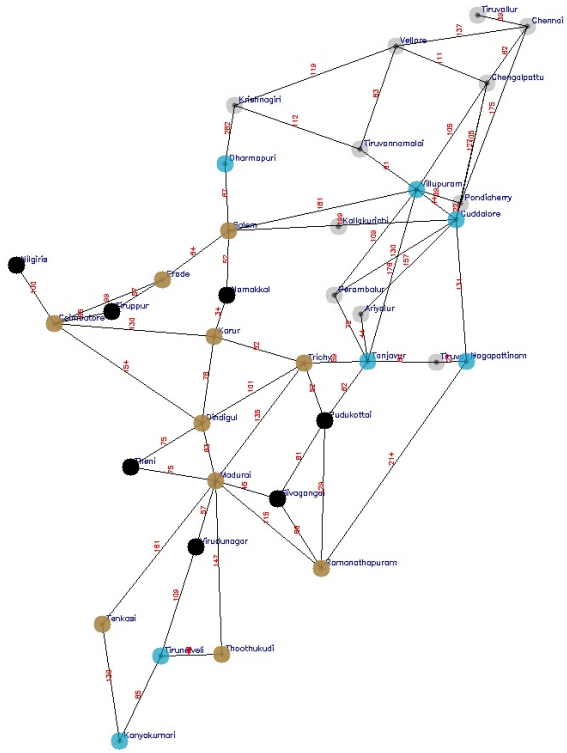
Fontier: ['Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram', 'Nagapattinam', 'Kanyakumari', 'Tirunelveli']



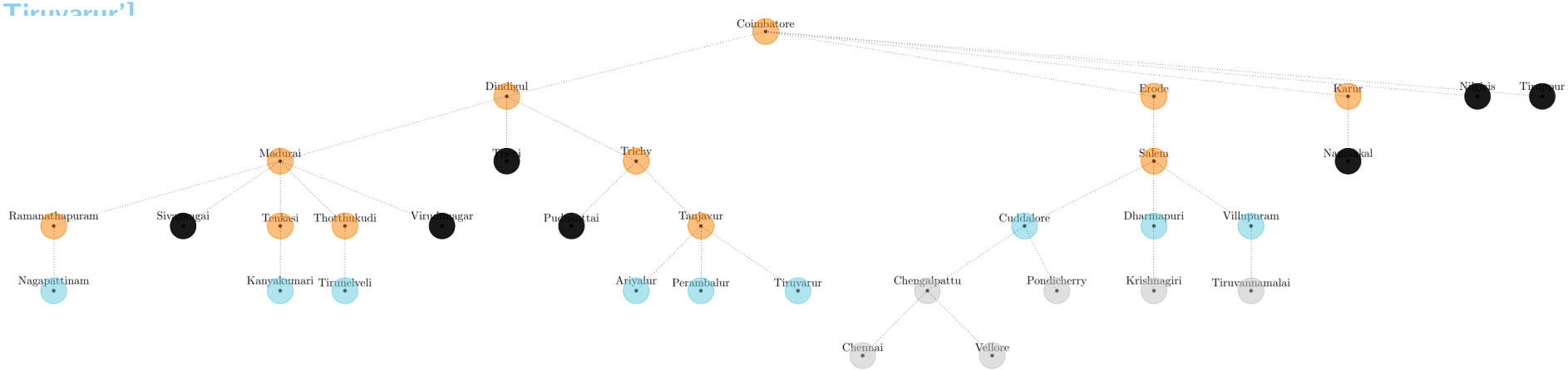


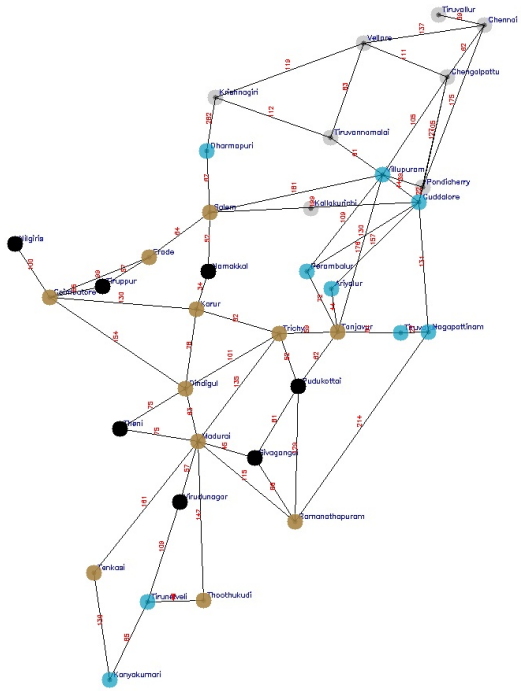
Fontier: ['Tanjavur', 'Cuddalore', 'Dharmapuri', 'Villupuram', 'Nagapattinam', 'Kanyakumari', 'Tirunelveli']



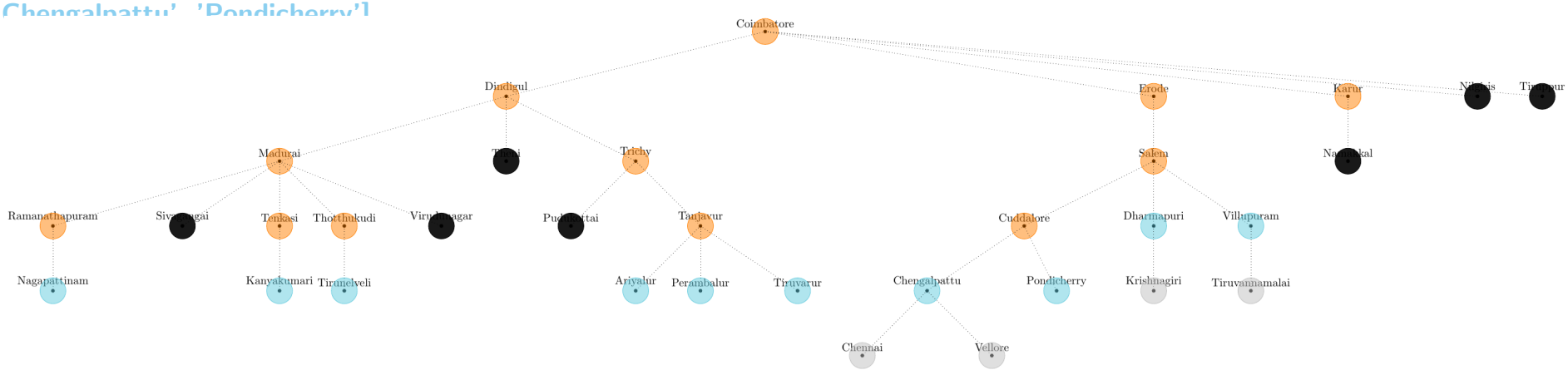


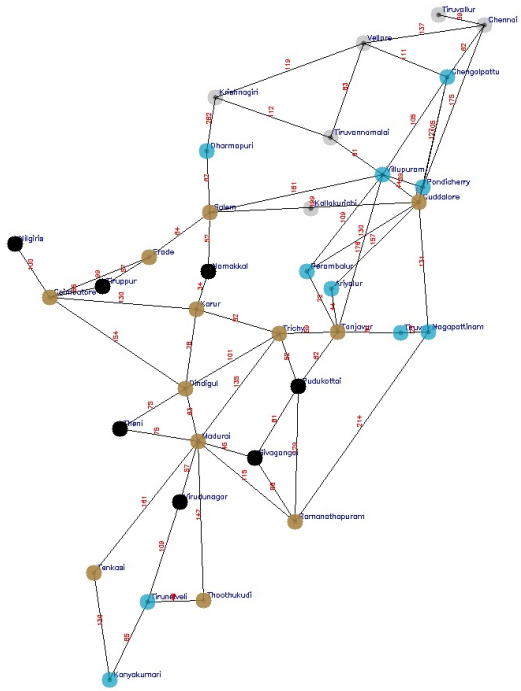
Fontier: ['Cuddalore', 'Dharmapuri', 'Villupuram', 'Nagapattinam', 'Kanyakumari', 'Tirunelveli', 'Ariyalur', 'Perambalur', 'Tiruvarur']



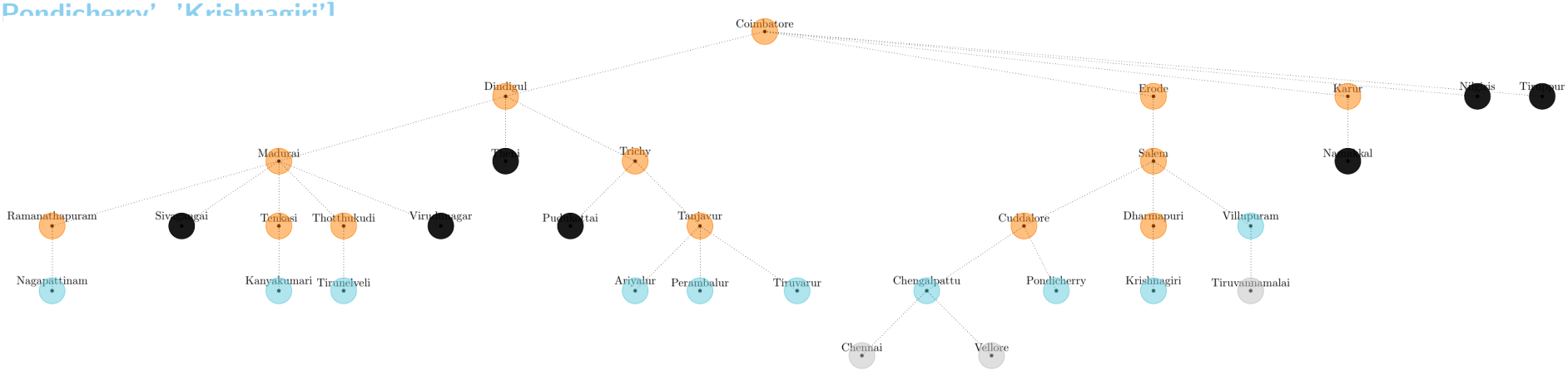


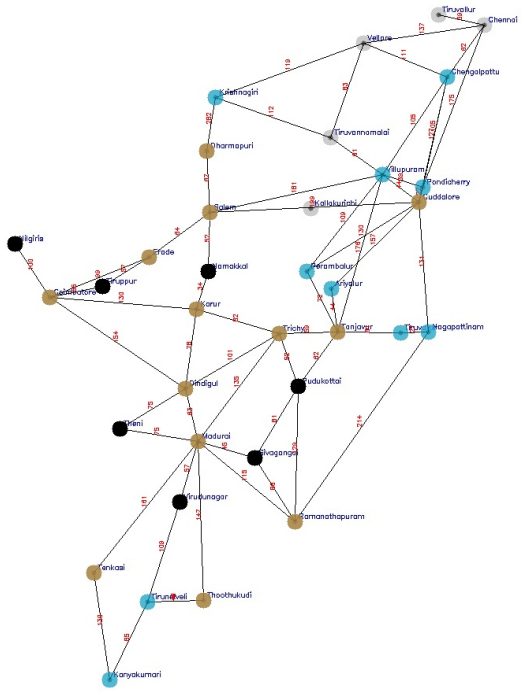
Fontier: ['Dharmapuri', 'Villupuram', 'Nagapattinam', 'Kanyakumari', 'Tirunelveli', 'Ariyalur', 'Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry']



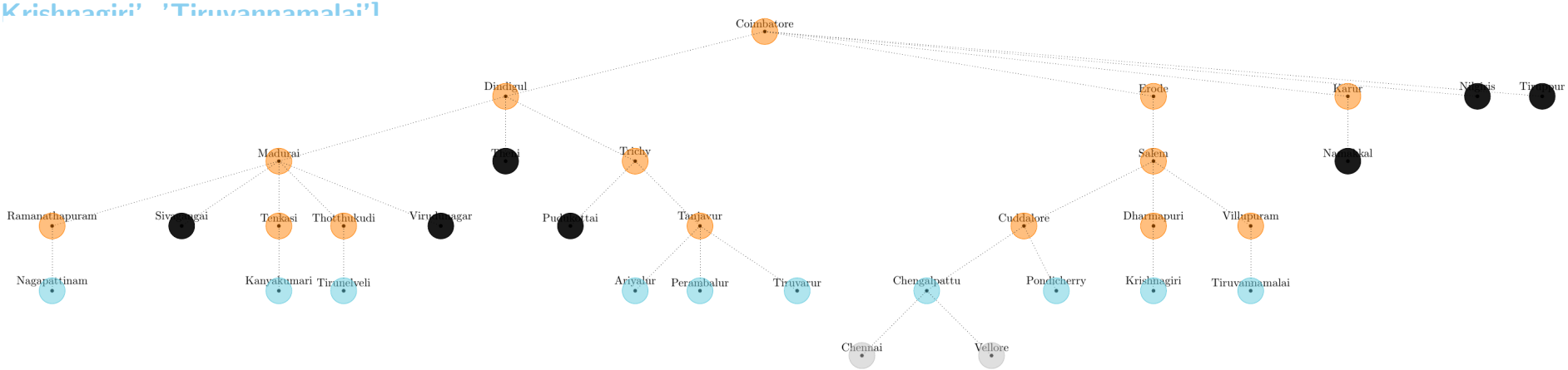


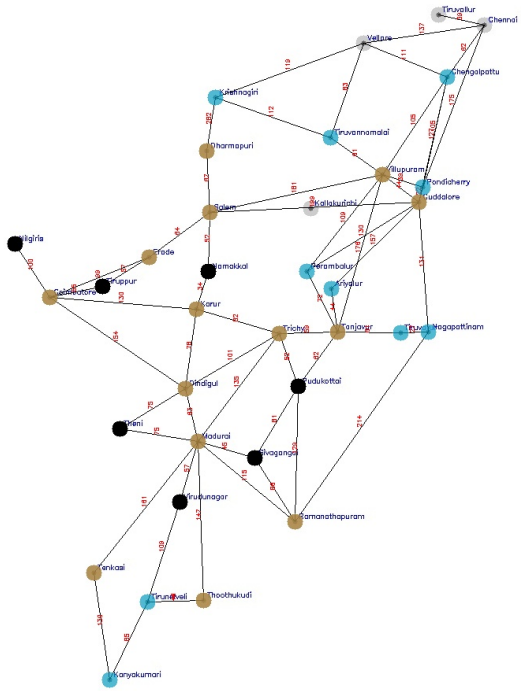
Fontier: ['Villupuram', 'Nagapattinam', 'Kanyakumari', 'Tirunelveli', 'Ariyalur', 'Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri']



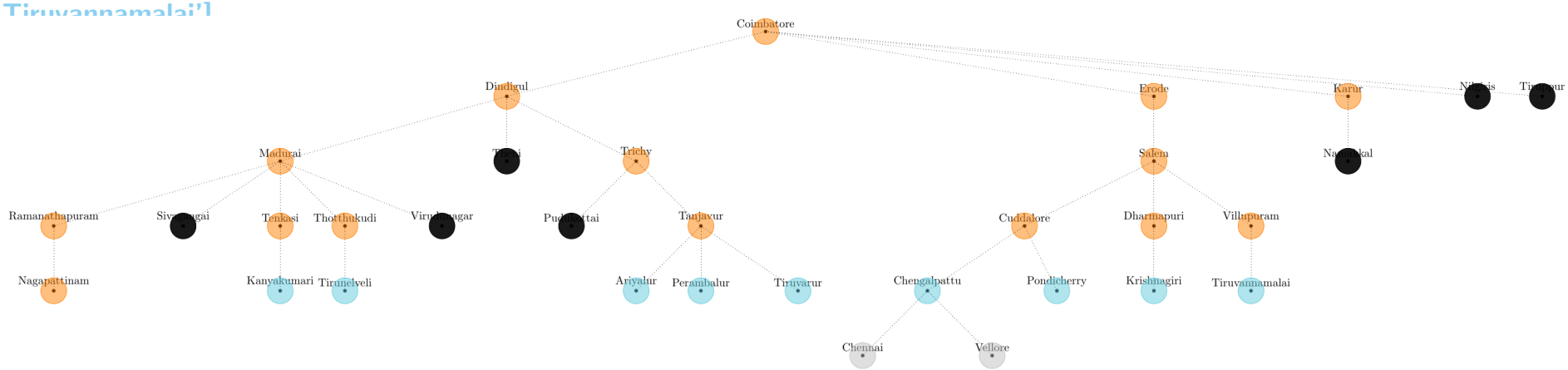


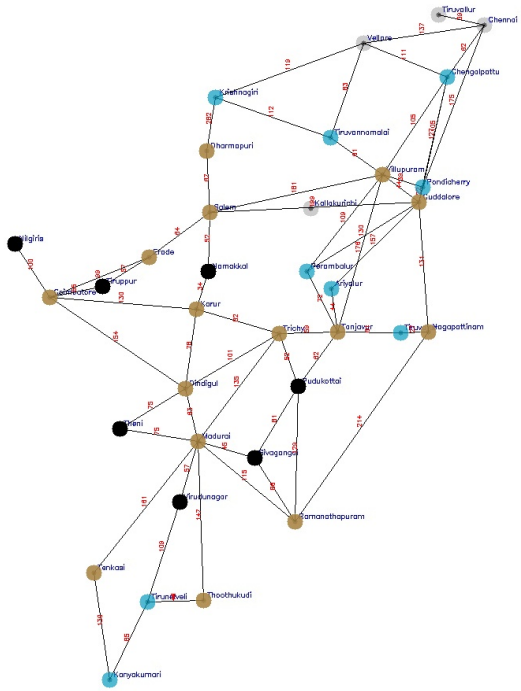
Fontier: ['Nagapattinam', 'Kanyakumari', 'Tirunelveli', 'Ariyalur', 'Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



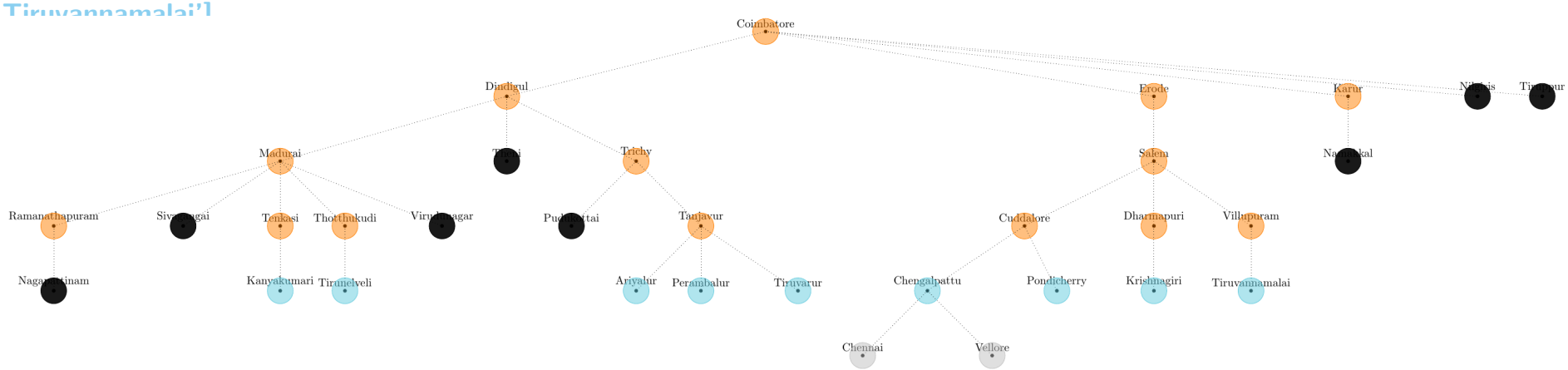


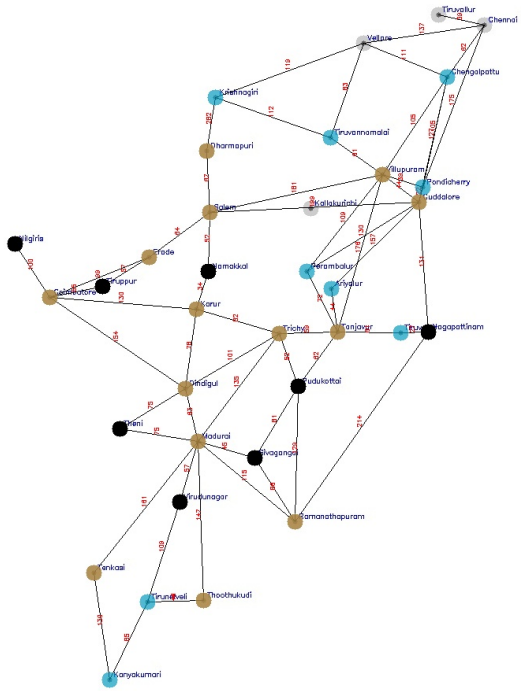
Fontier: ['Kanyakumari', 'Tirunelveli', 'Ariyalur', 'Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



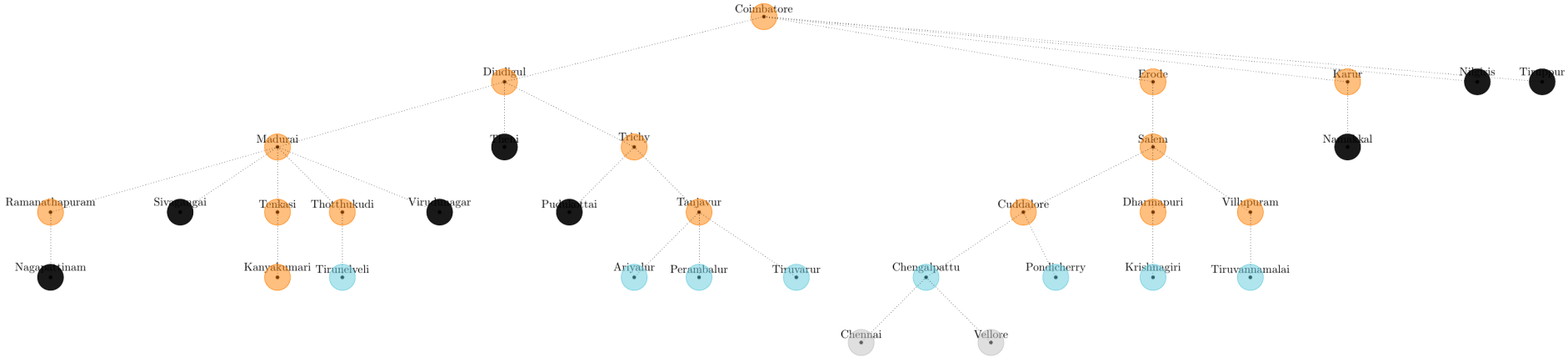


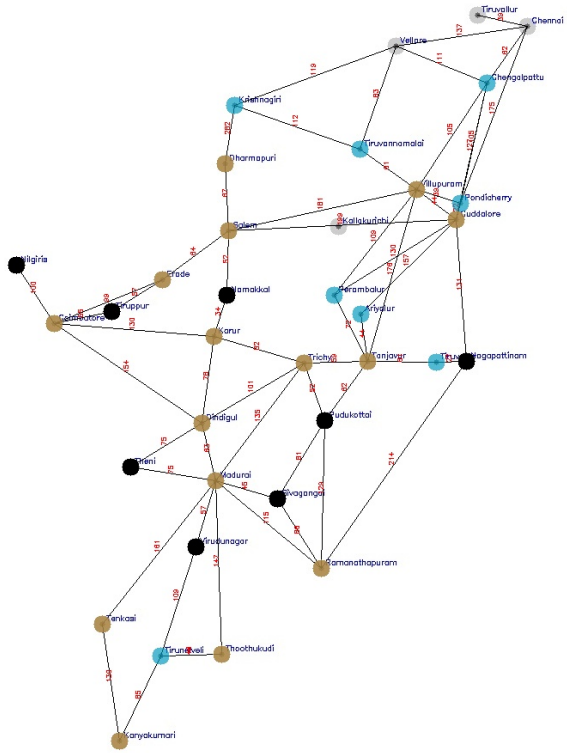
Fontier: ['Kanyakumari', 'Tirunelveli', 'Ariyalur', 'Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



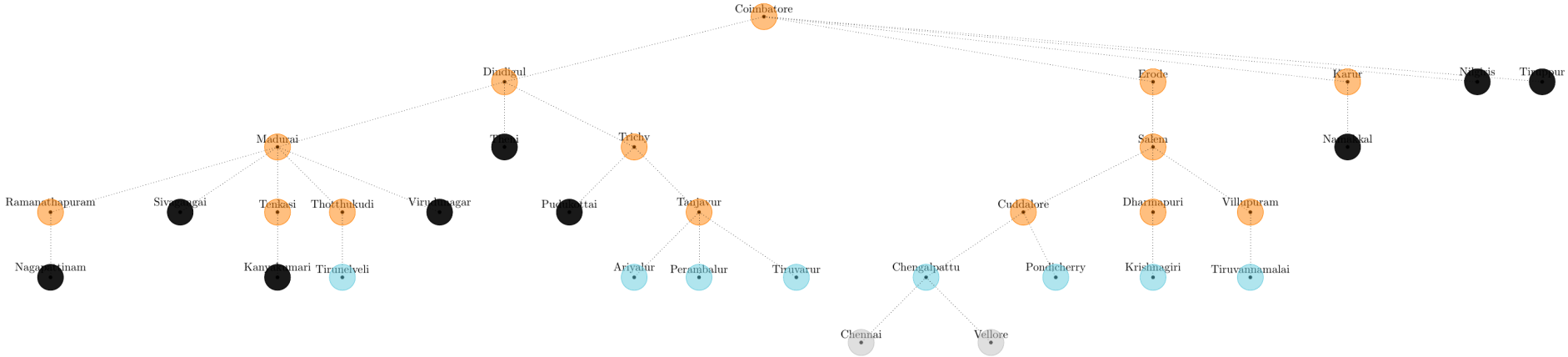


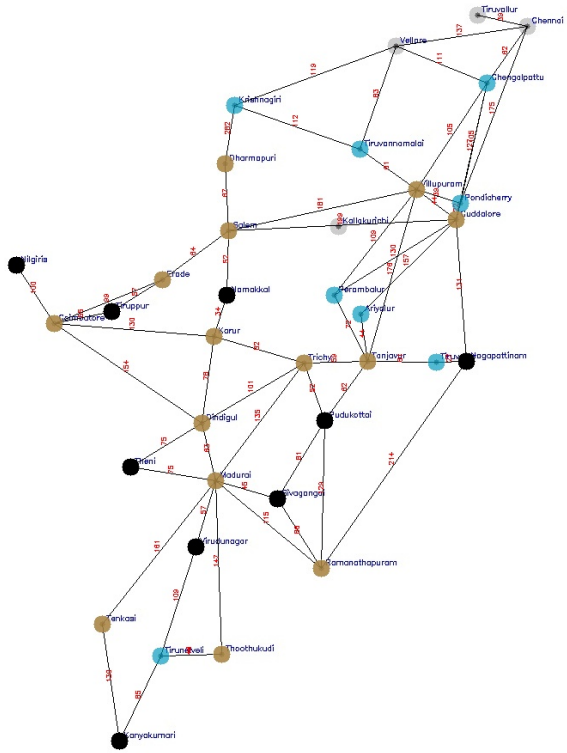
Fontier: ['Tirunelveli', 'Ariyalur', 'Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



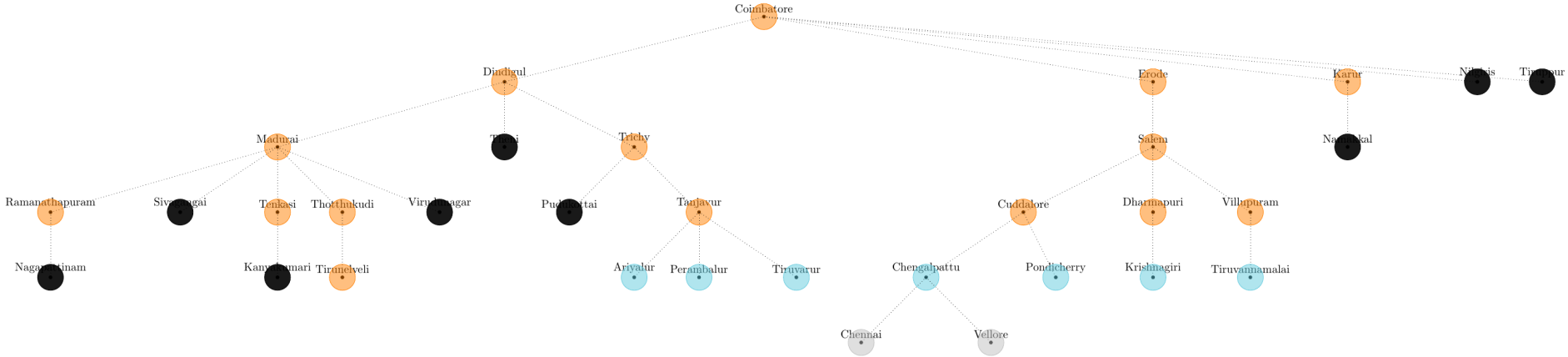


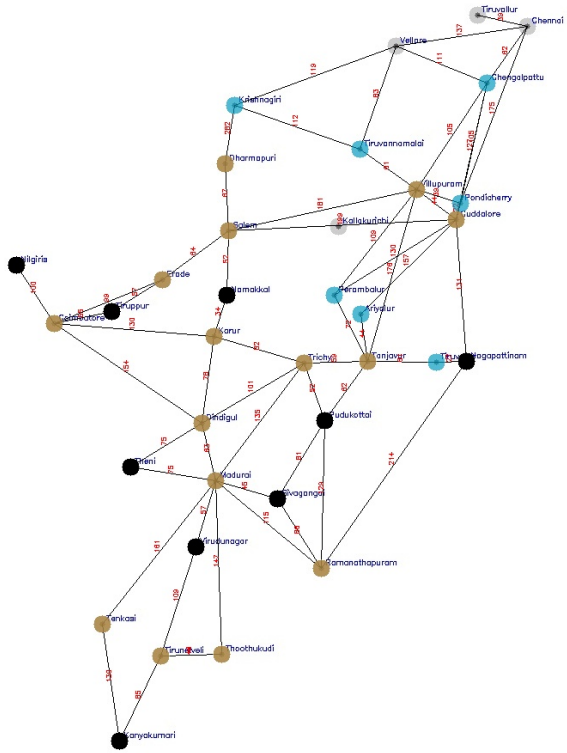
Fontier: ['Tirunelveli', 'Ariyalur', 'Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



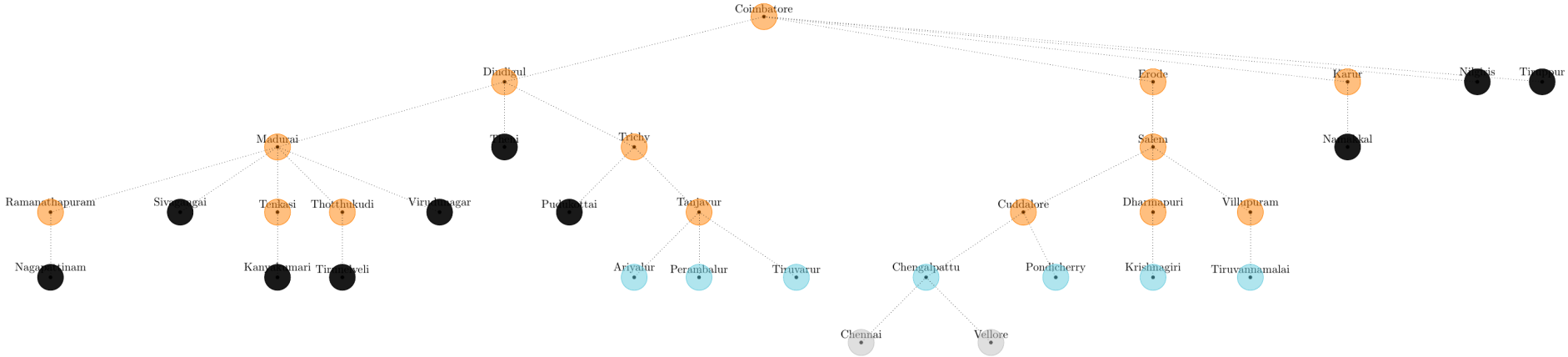


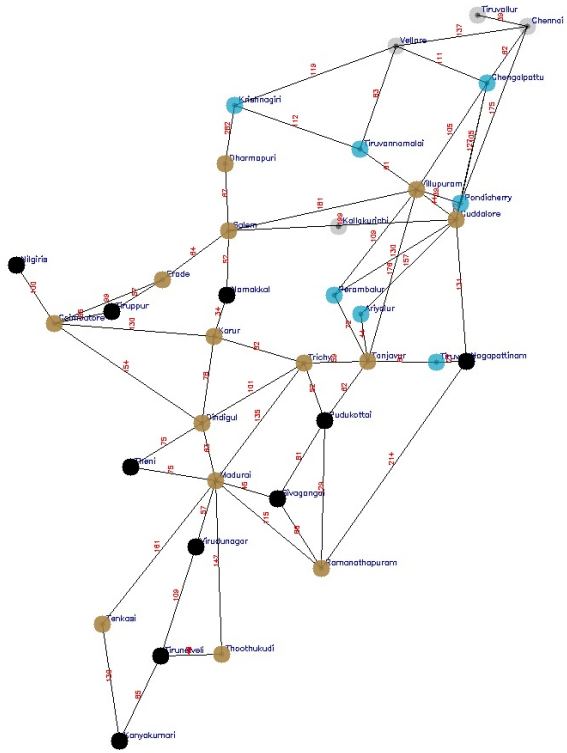
Fontier: ['Ariyalur', 'Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



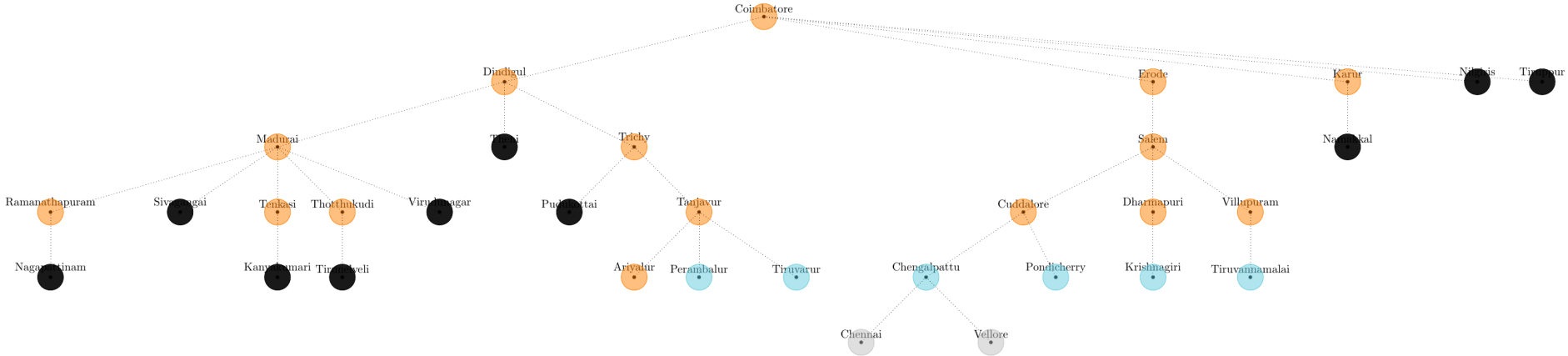


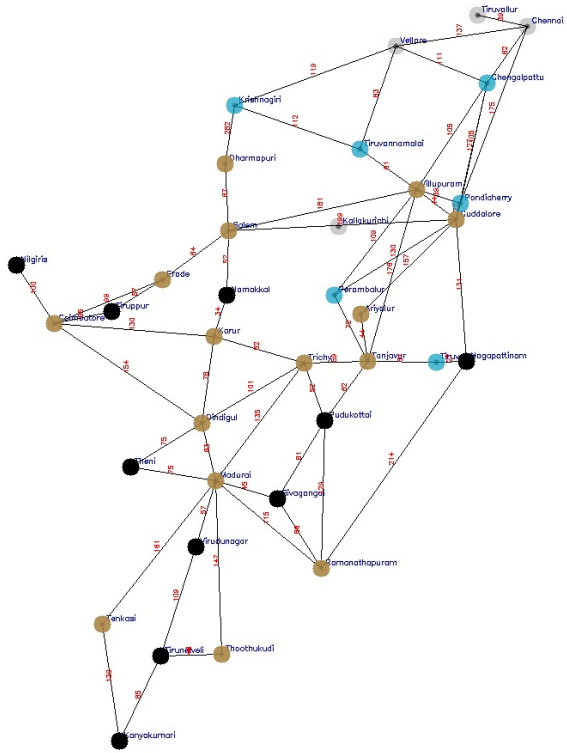
Fontier: ['Ariyalur', 'Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



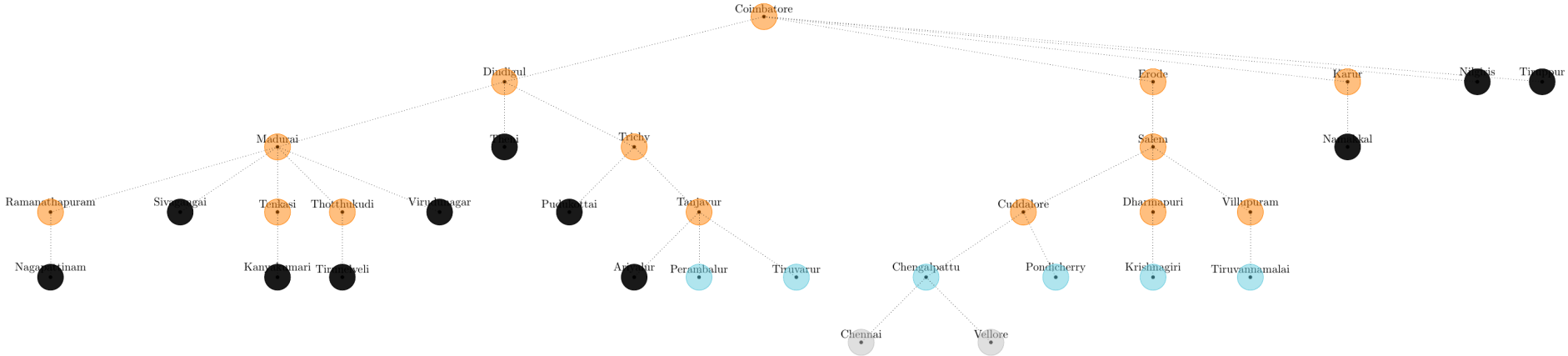


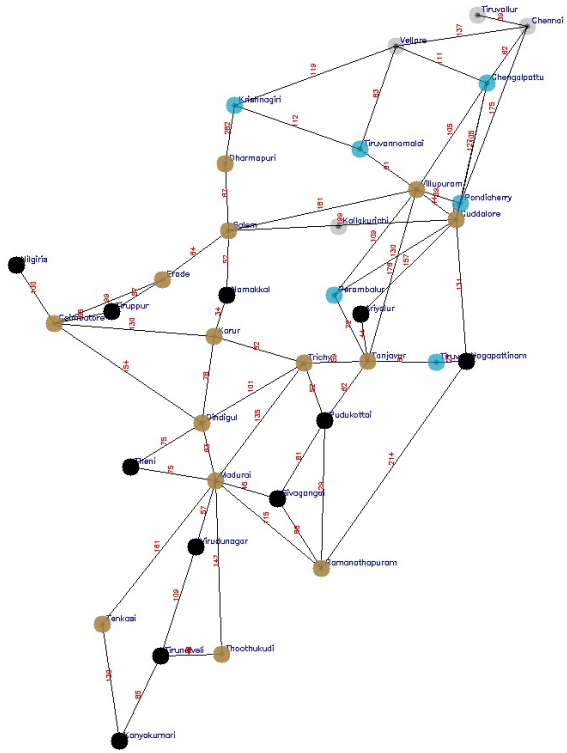
Fontier: ['Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



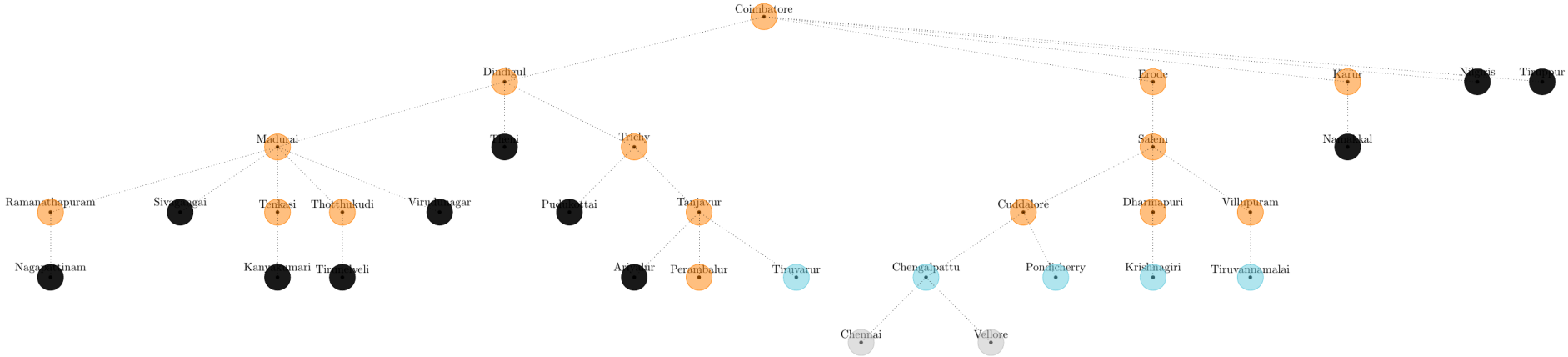


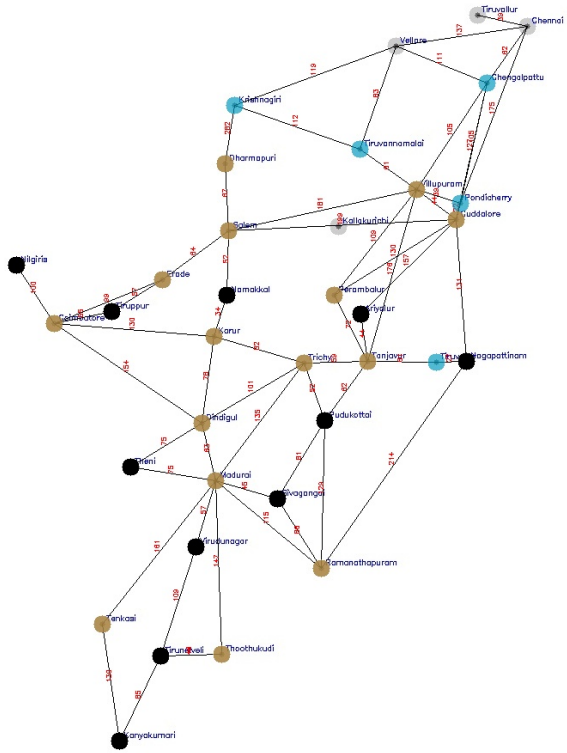
Fontier: ['Perambalur', 'Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



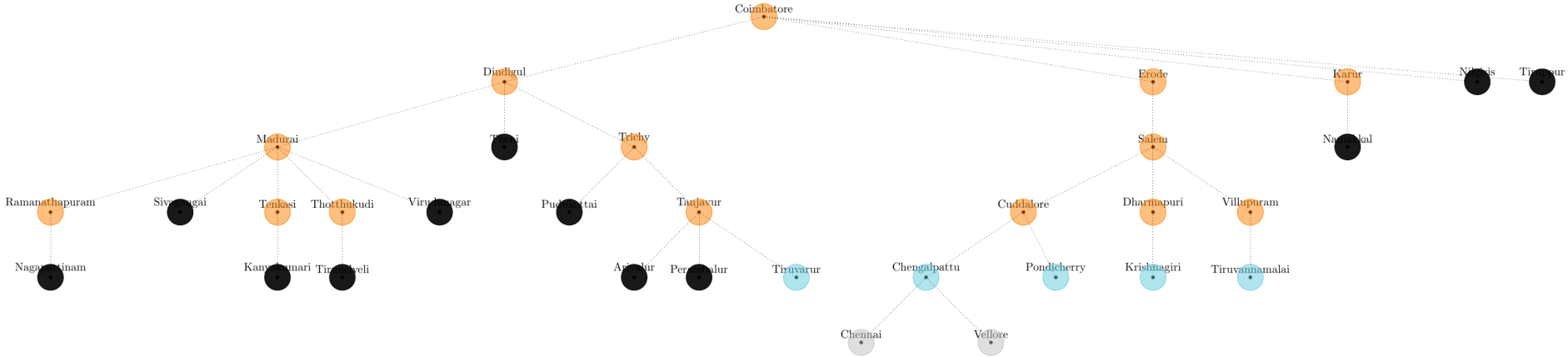


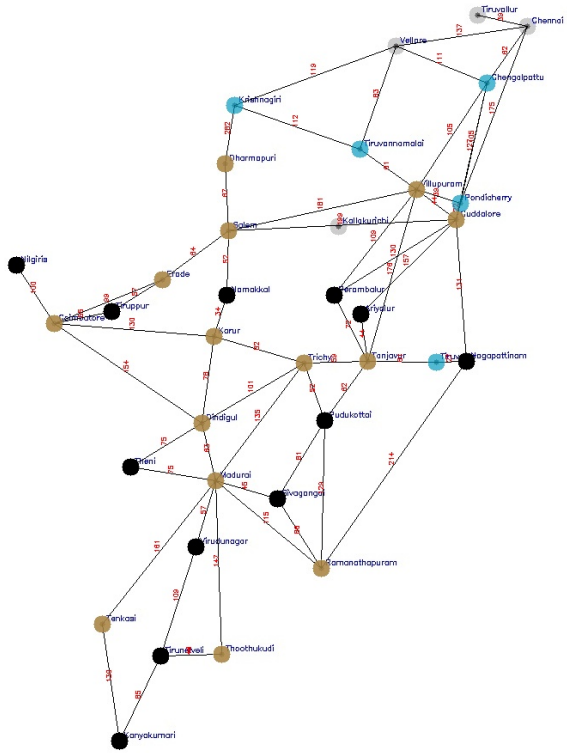
Fontier: ['Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



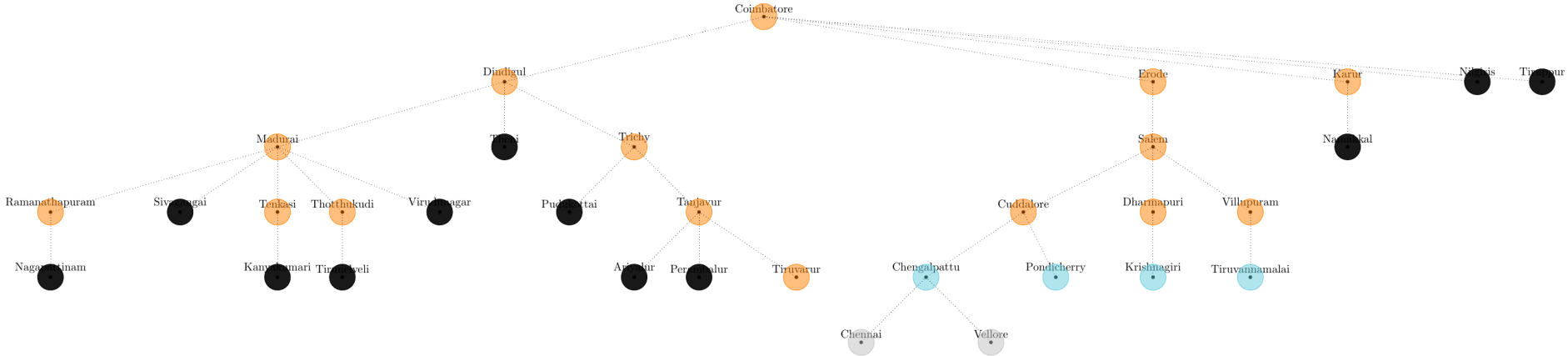


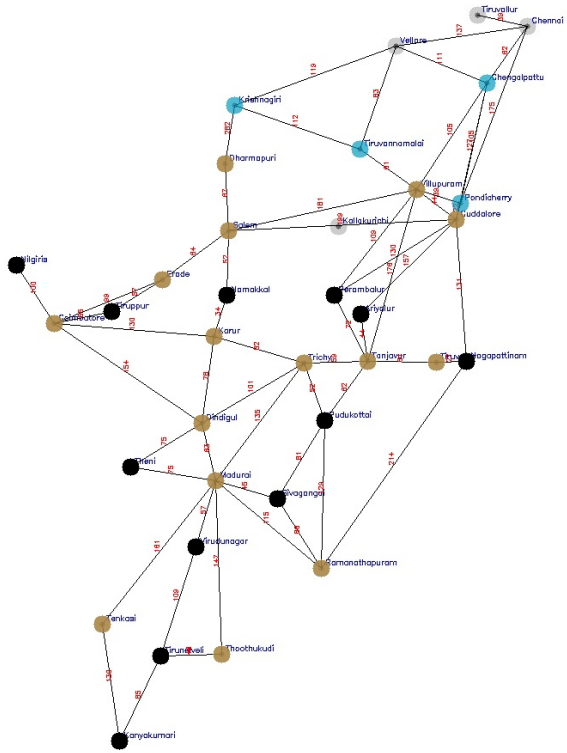
Fontier: ['Tiruvarur', 'Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



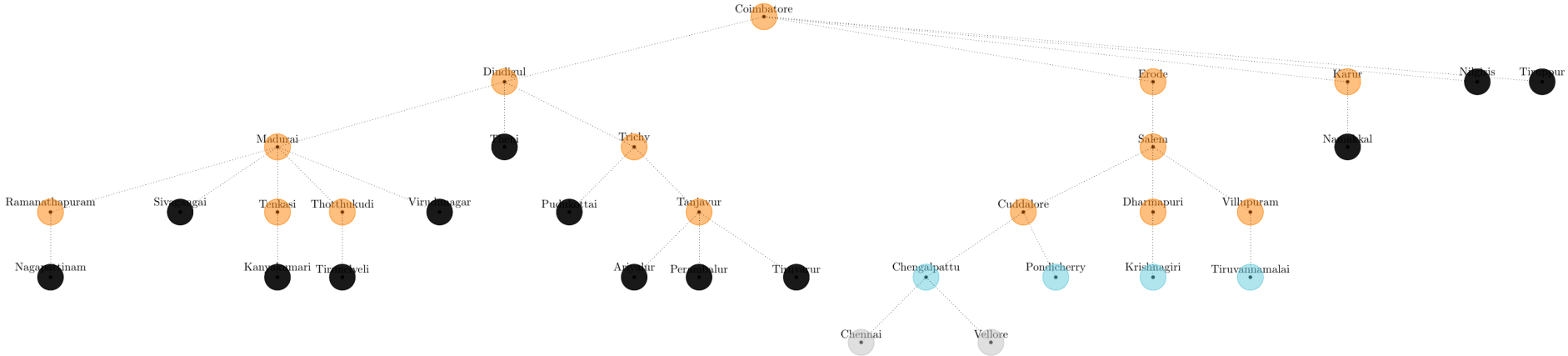


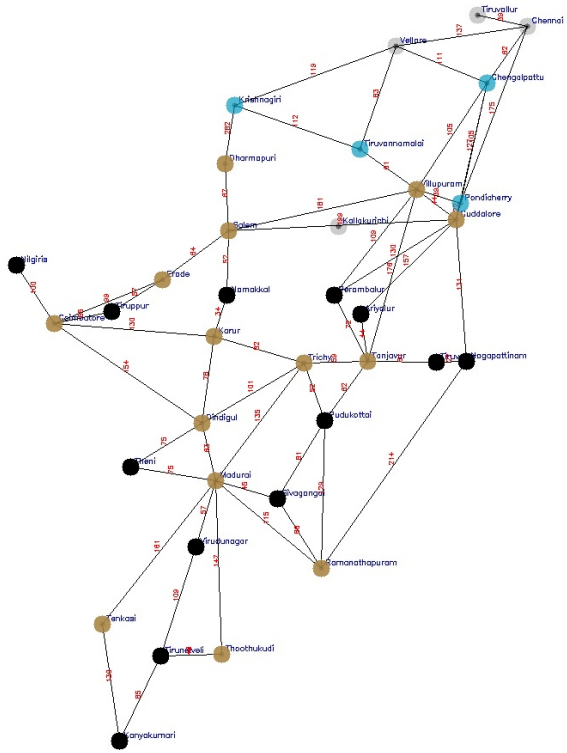
Fontier: ['Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



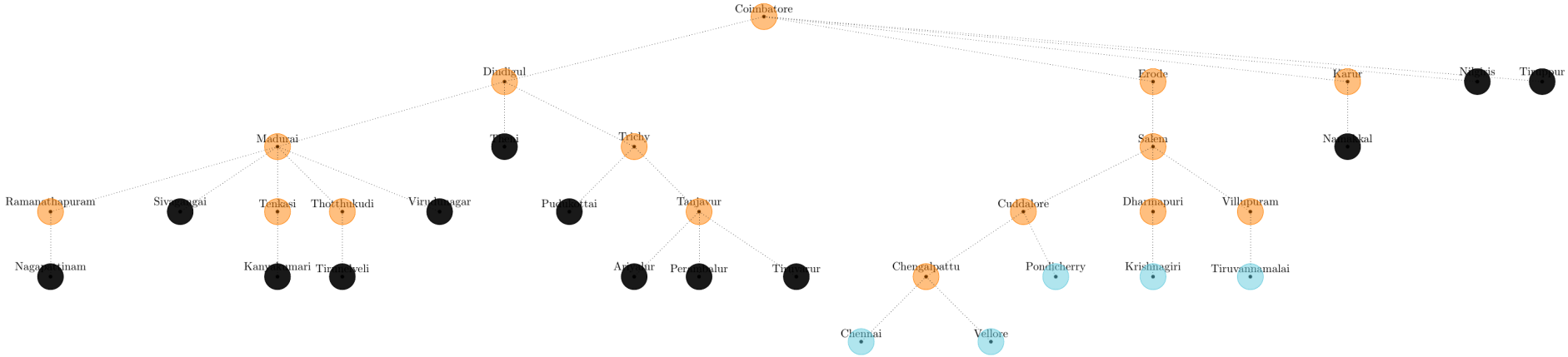


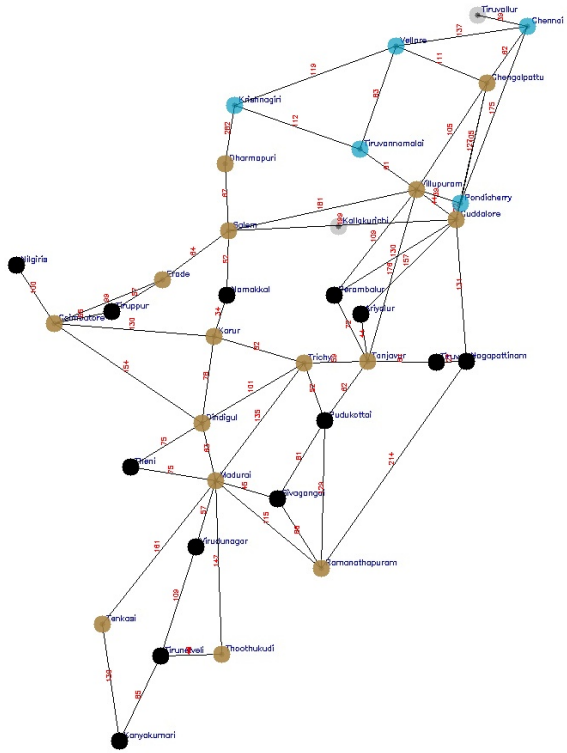
Fontier: ['Chengalpattu', 'Pondicherry', 'Krishnagiri', 'Tiruvannamalai']



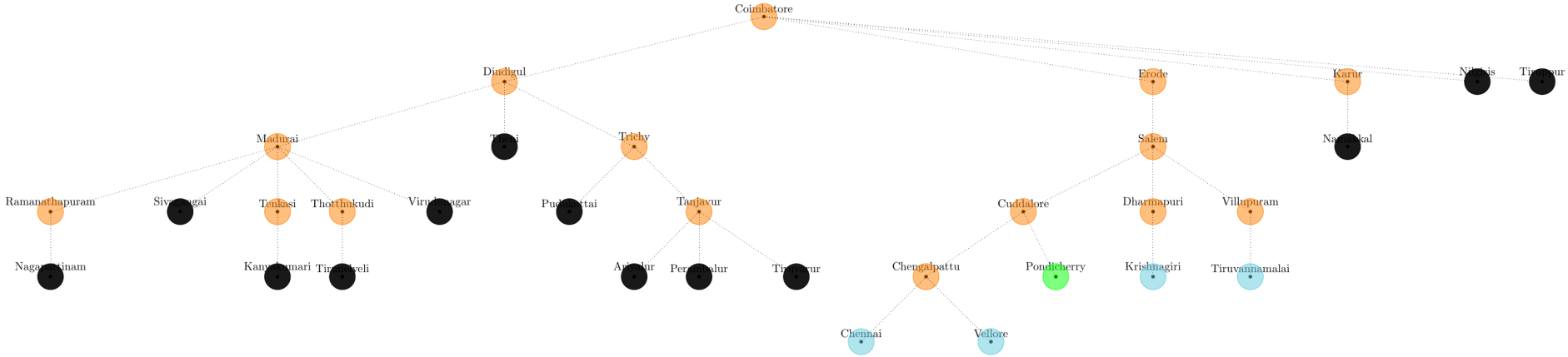


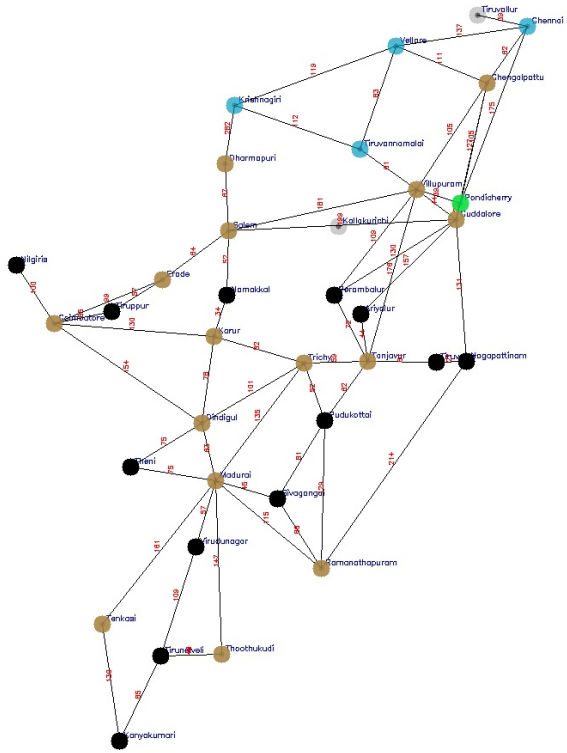
Fontier: ['Pondicherry', 'Krishnagiri', 'Tiruvannamalai', 'Chennai', 'Vellore']



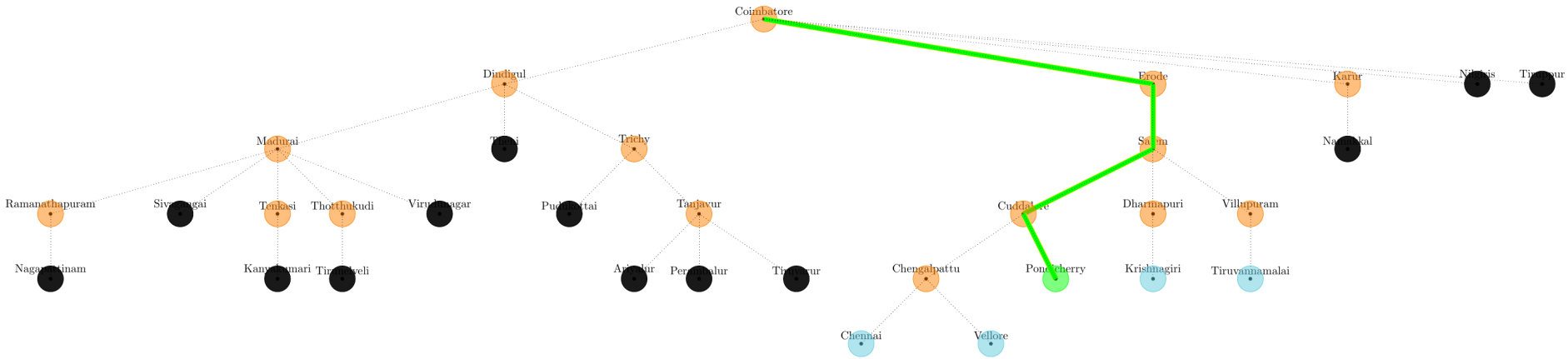


Fontier: ['Krishnagiri', 'Tiruvannamalai', 'Chennai', 'Vellore']





The path BFS took is from Coimbatore to Pondicherry is Coimbatore $->$ Erode $->$ Salem $->$ Cuddalore $->$ Pondicherry.

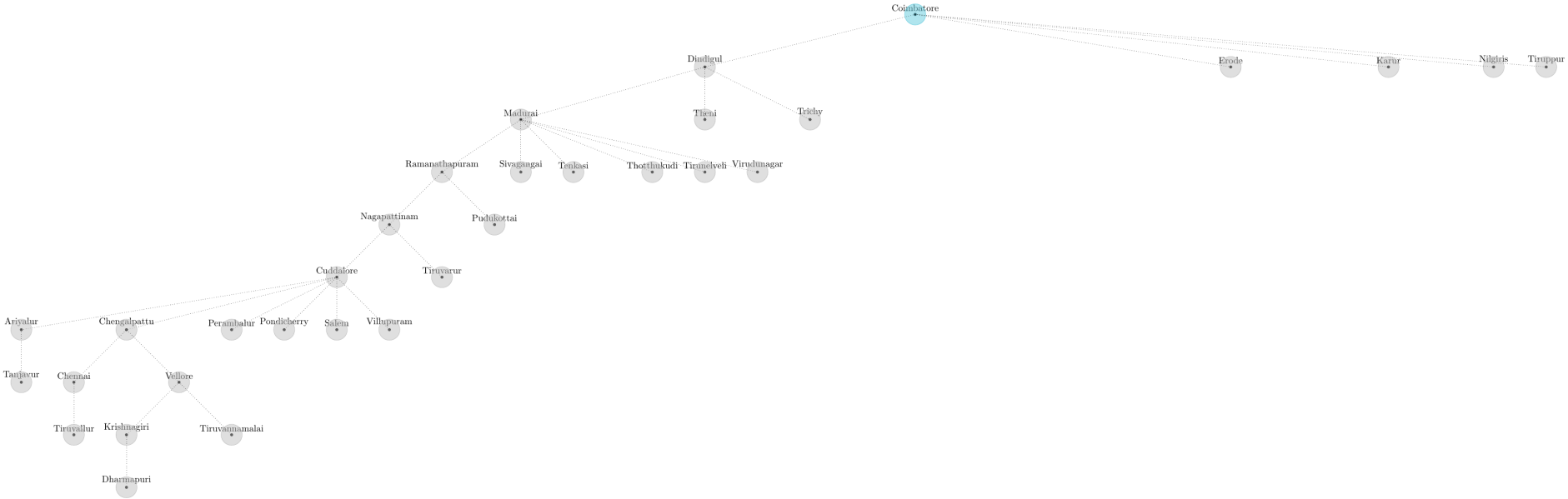


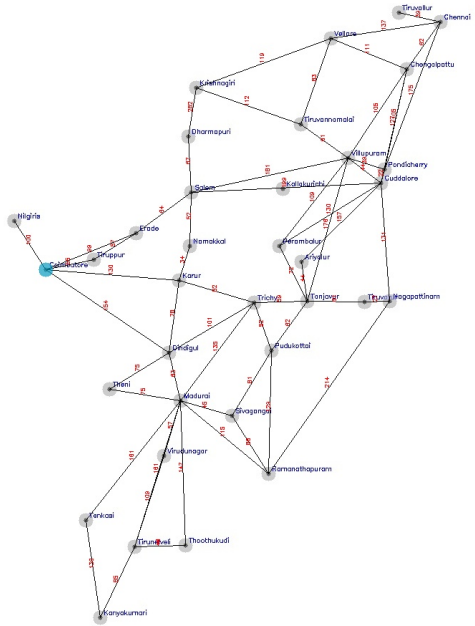
Depth First Search - DFS

- In DFS, we expand the deepest node first.
- It doesn't use any domain information.
- The data structure used is a **STACK**.

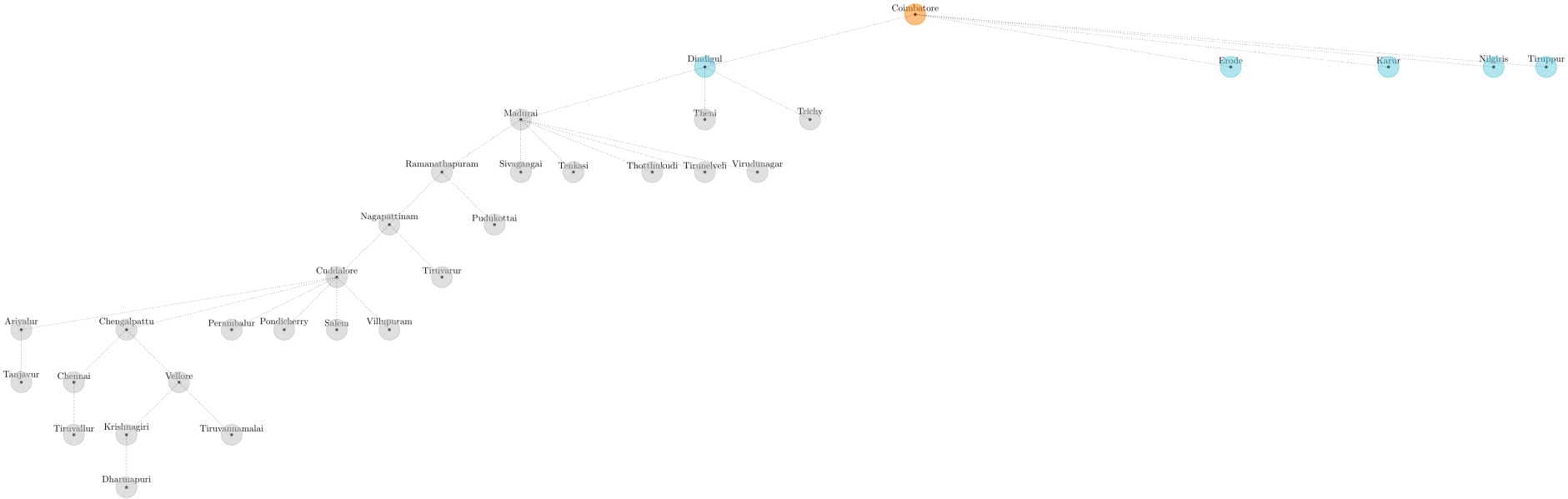
```
function DEPTH-FIRST-SEARCH(initialState, goalTest)
  returns SUCCESS or FAILURE:

  frontier = Stack.new(initialState)
  explored = Set.new()

  while not frontier.isEmpty()
    state = frontier.pop()
    explored.add(state)
    if goalTest(state):
      return SUCCESS(state)

    for neighbor in state.neighbors():
      if neighbor not in frontier U explored:
        frontier.push(neighbor)
  return FAILURE
```

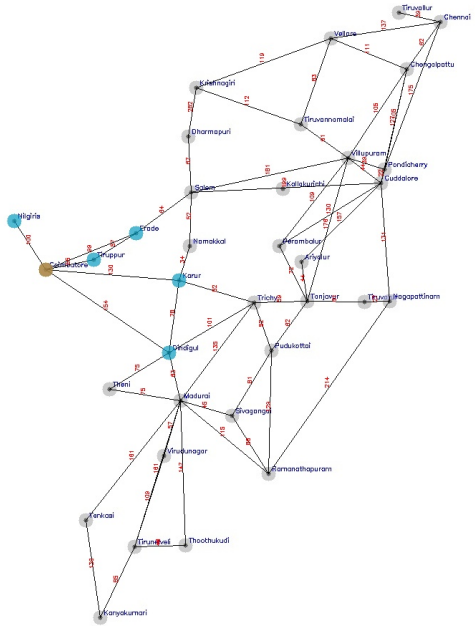
Fontier: ['Coimbatore']



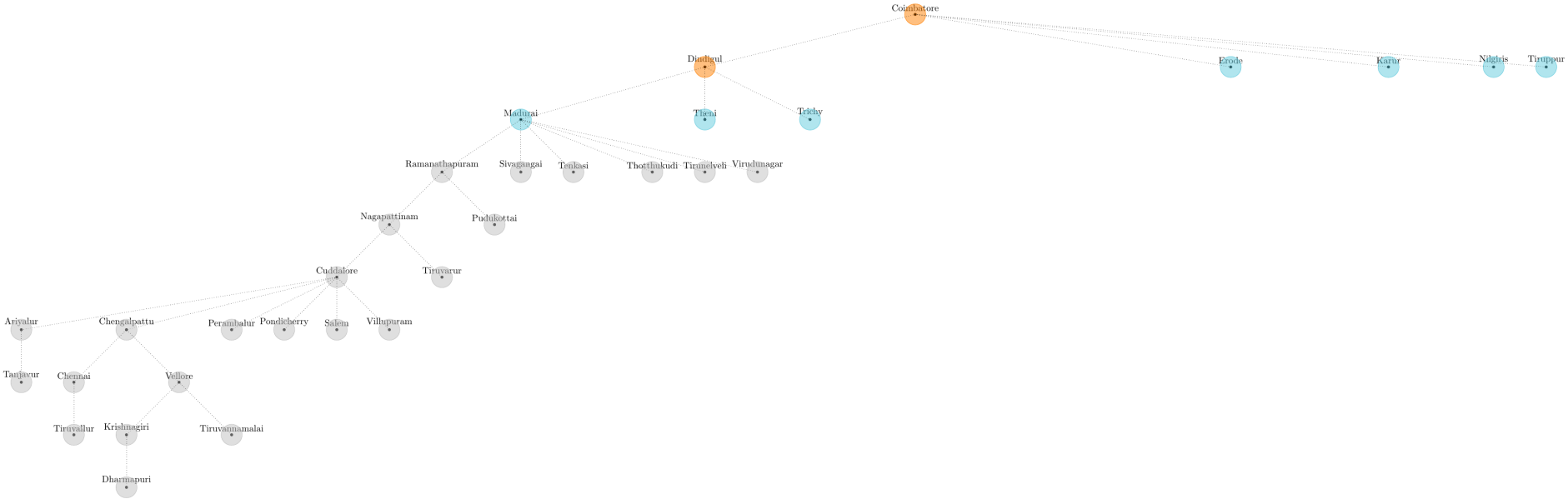


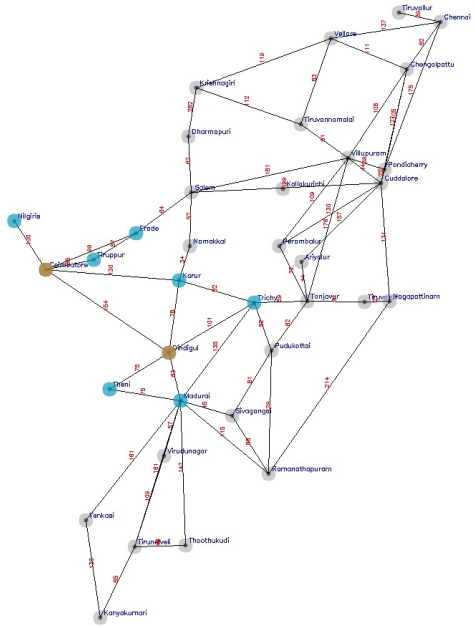
Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Dindigul']





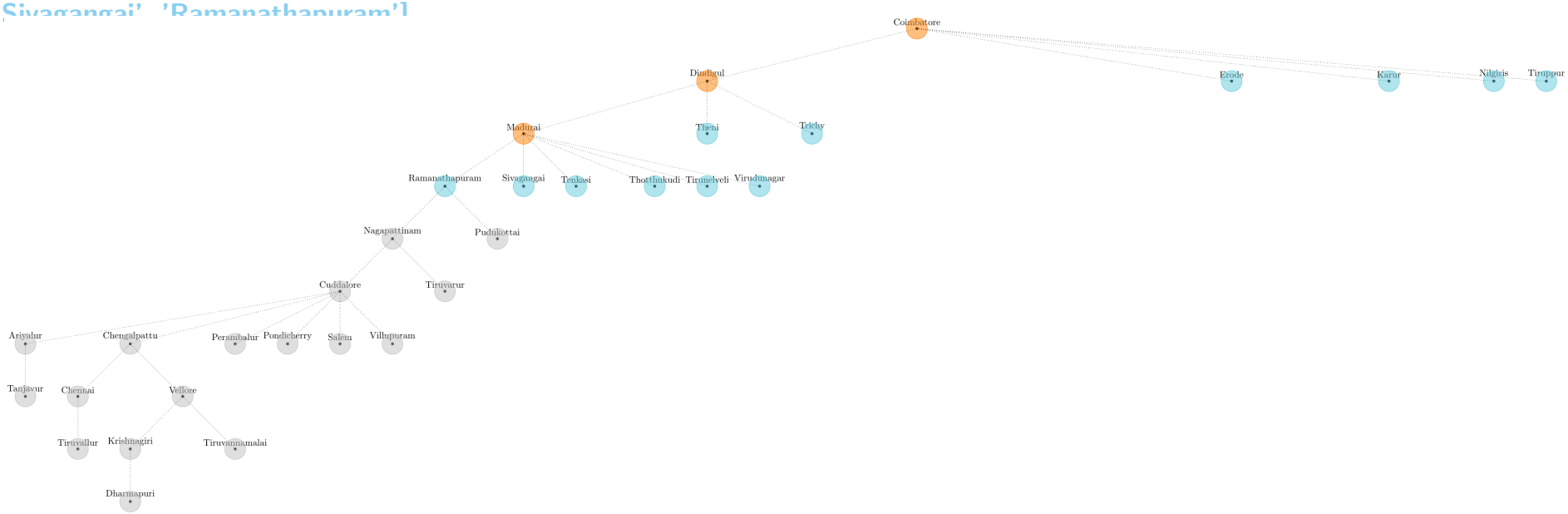
Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Madurai']

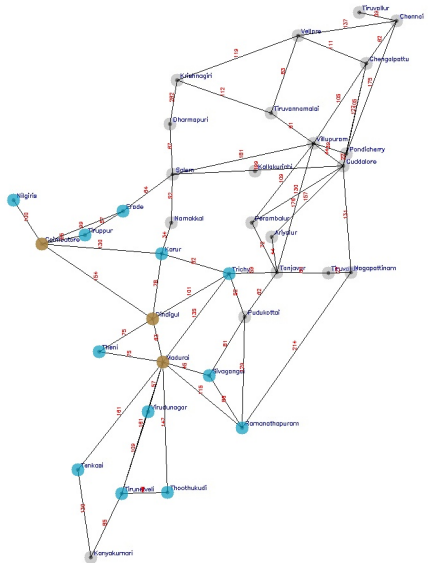




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

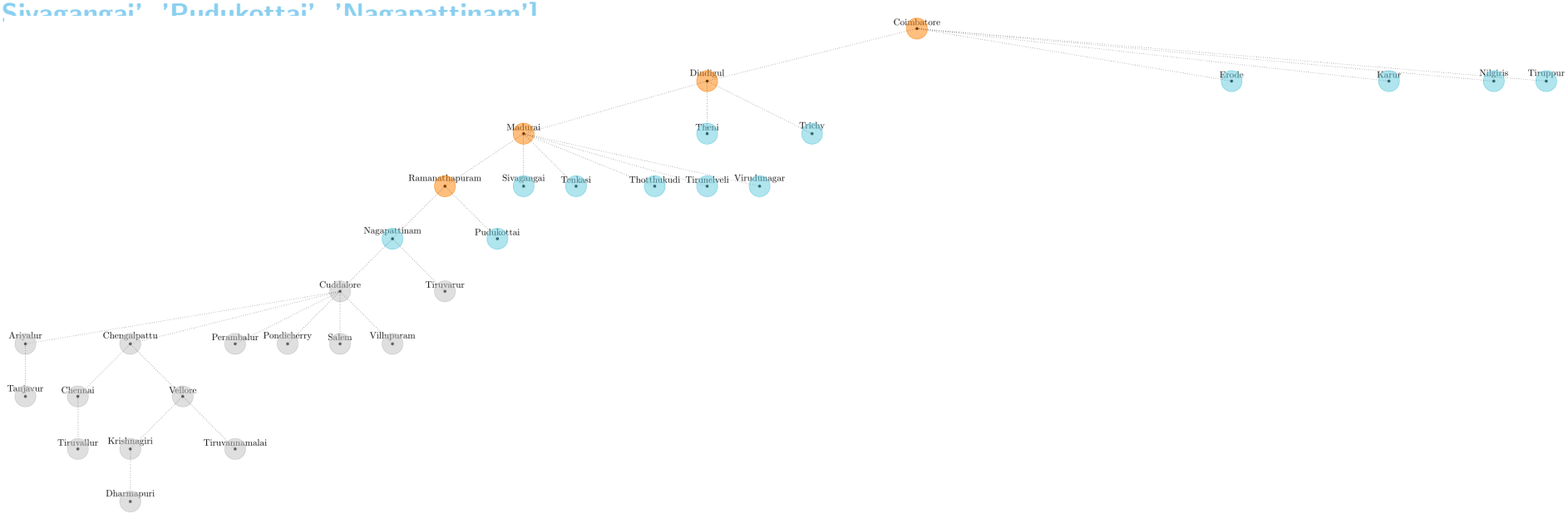
'Sivagangai', 'Ramanathapuram']

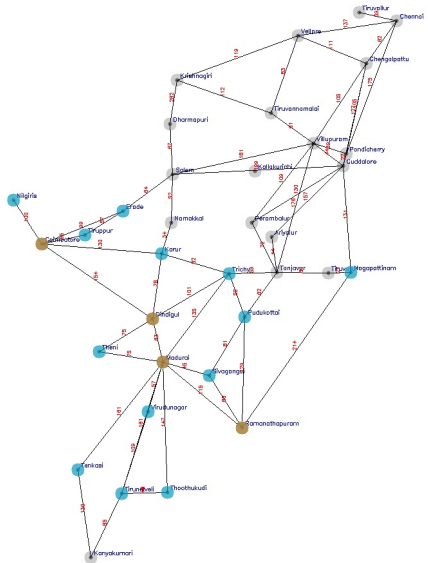




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

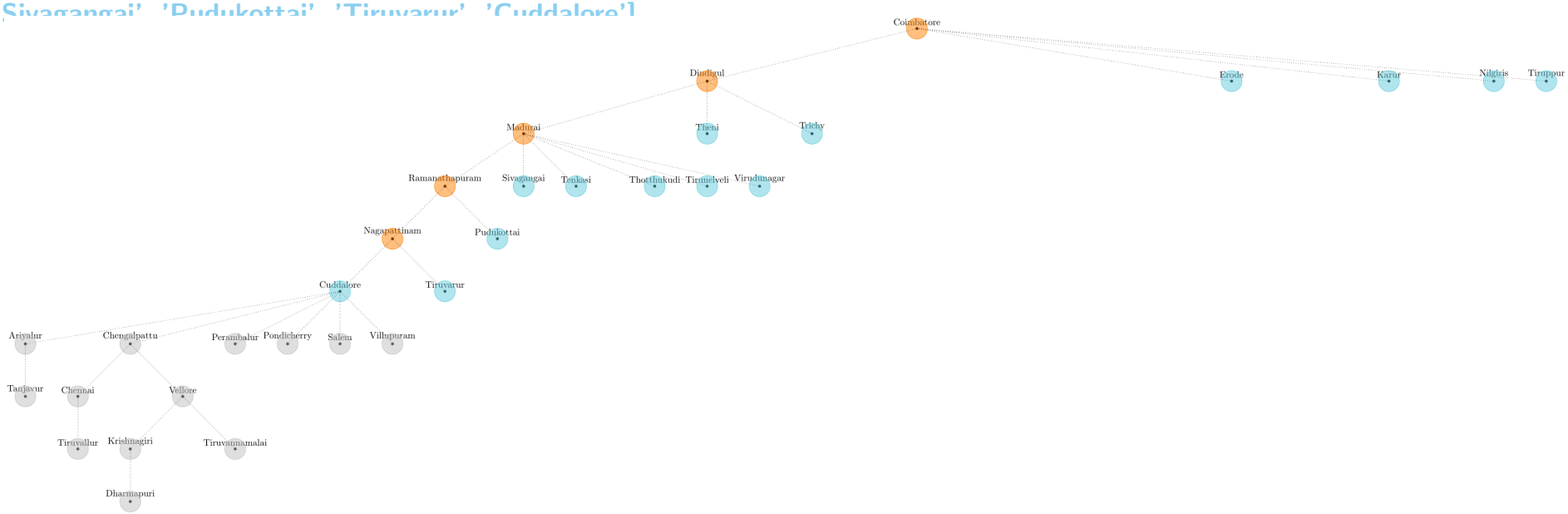
'Sivagangai', 'Pudukottai', 'Nagapattinam']

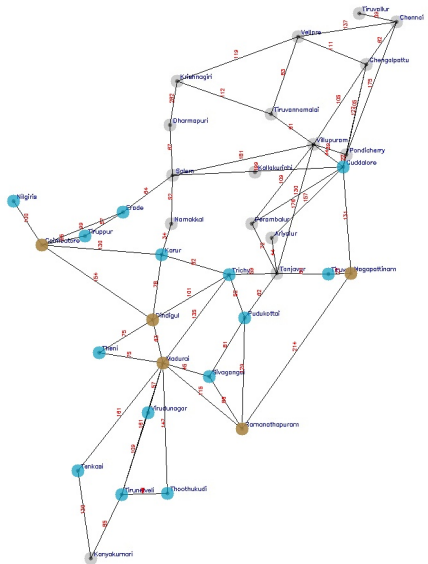




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

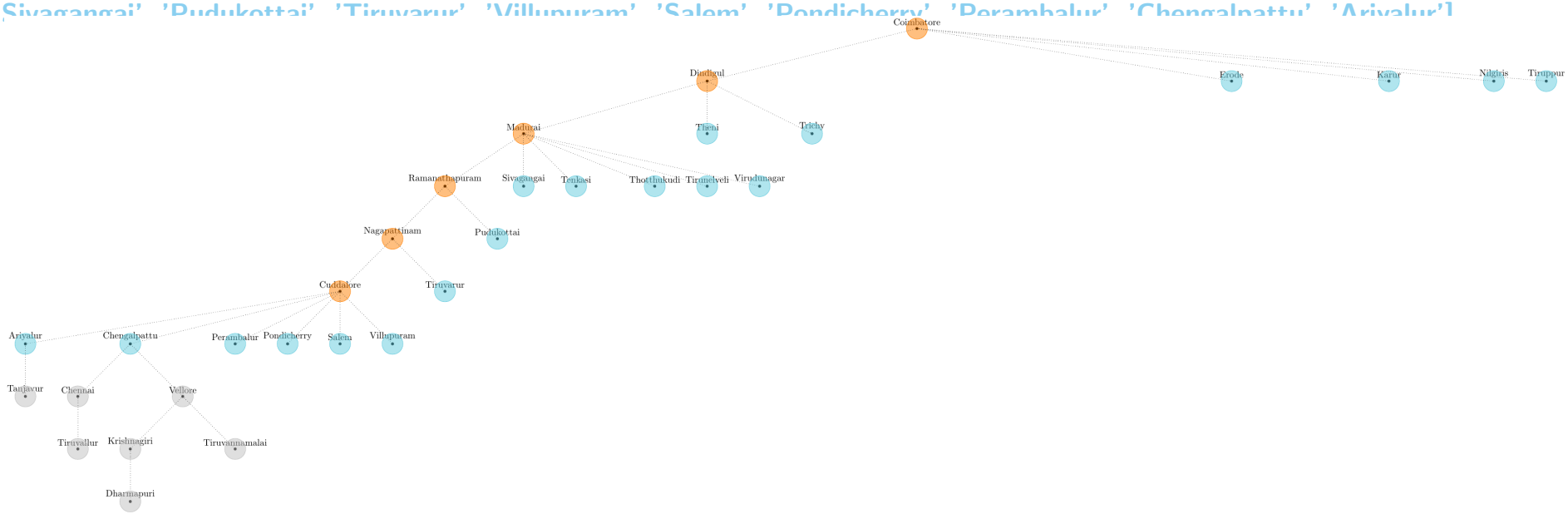
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Cuddalore']

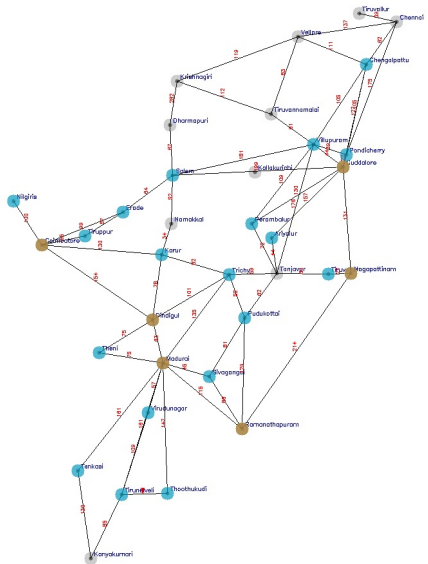




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

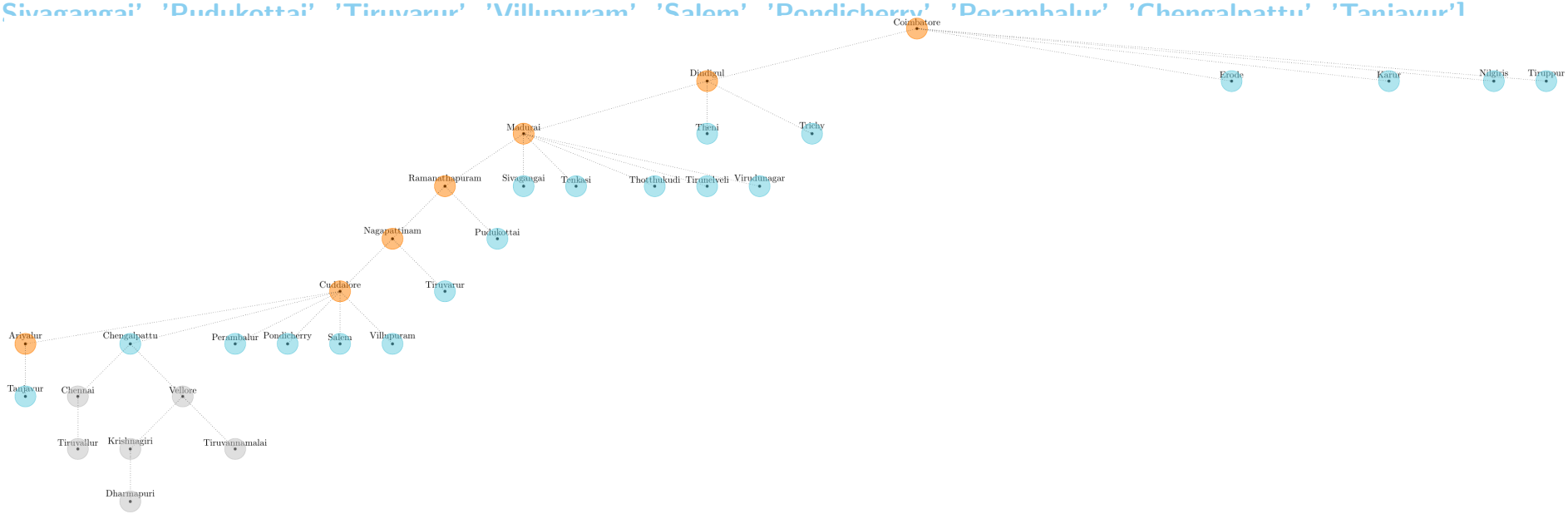
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Chengalpattu', 'Arivalur']

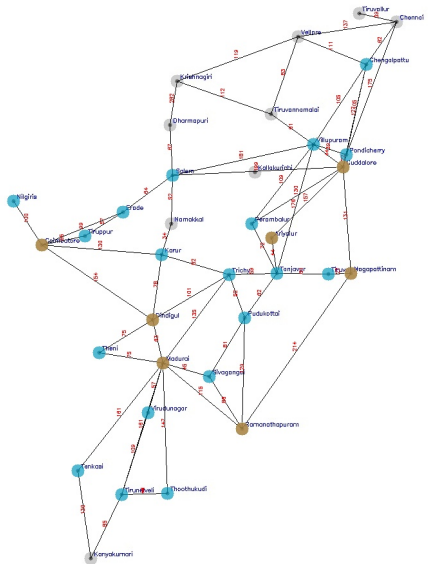




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

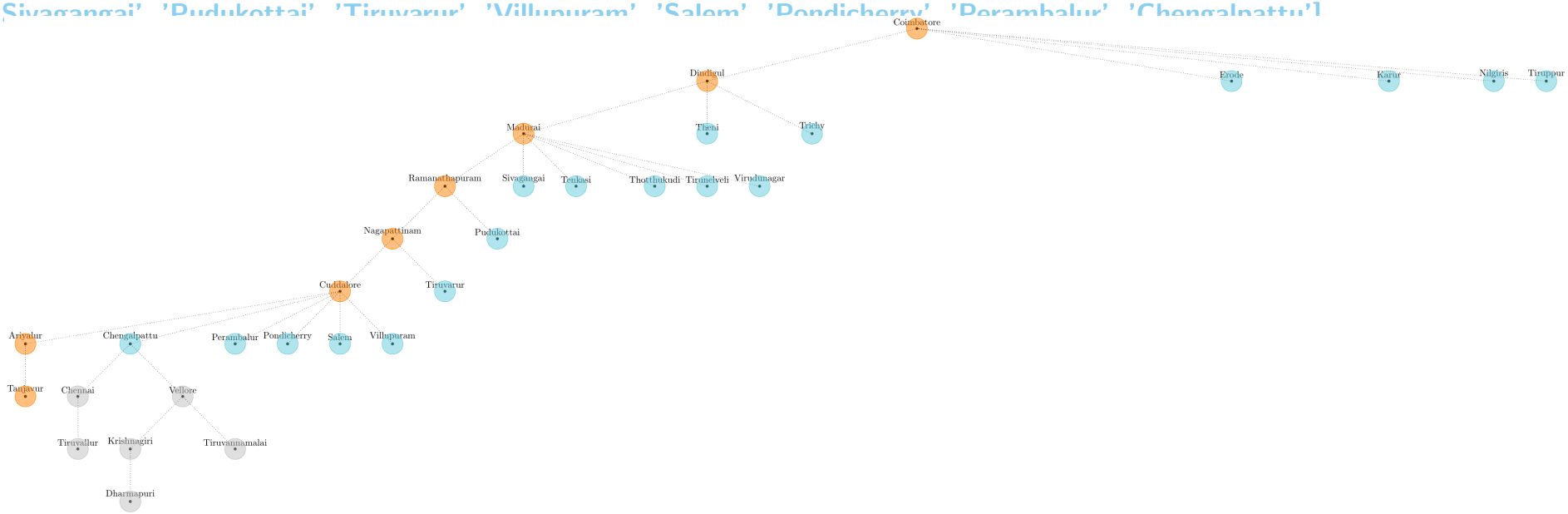
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Chengalpattu', 'Tanjavur']

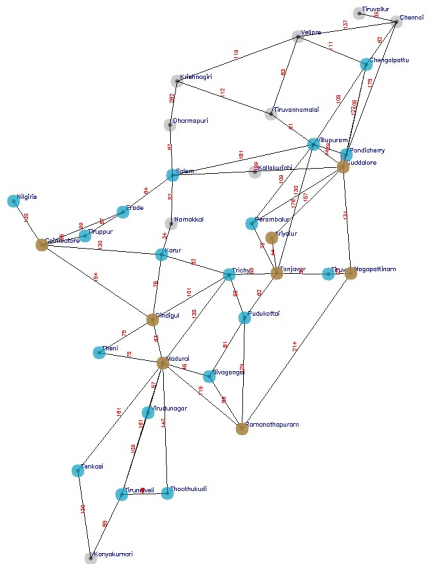




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

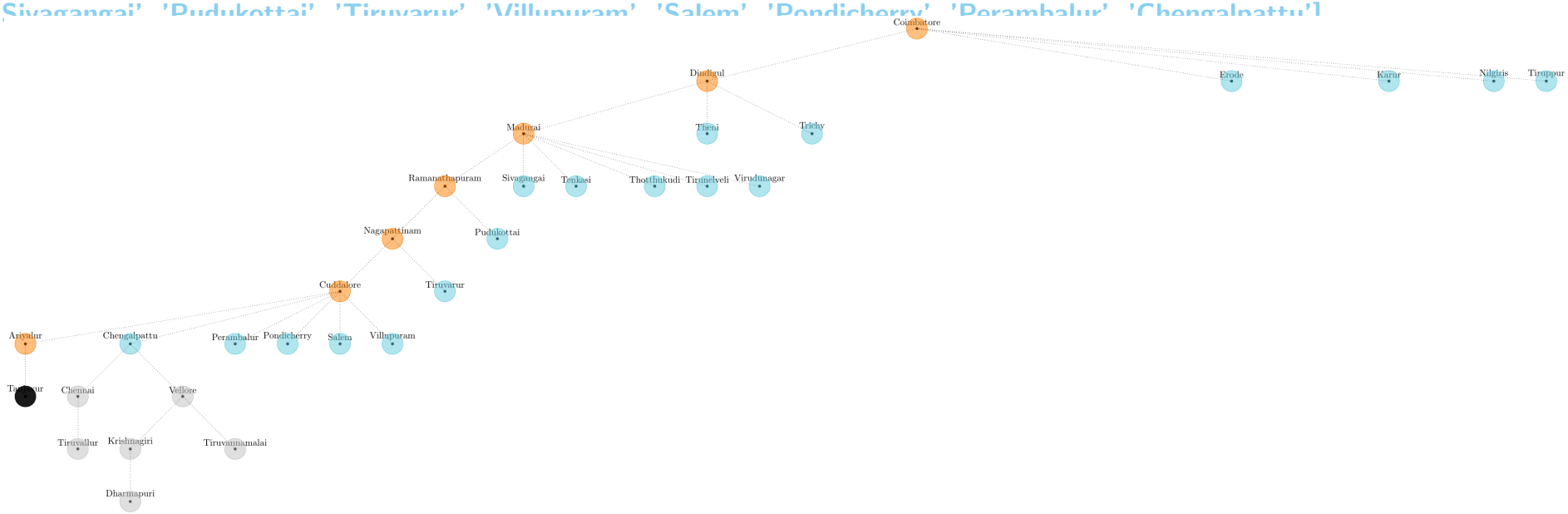
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Chengalpattu']

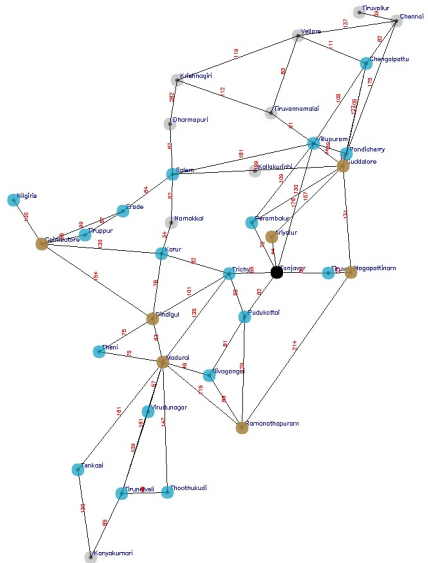




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

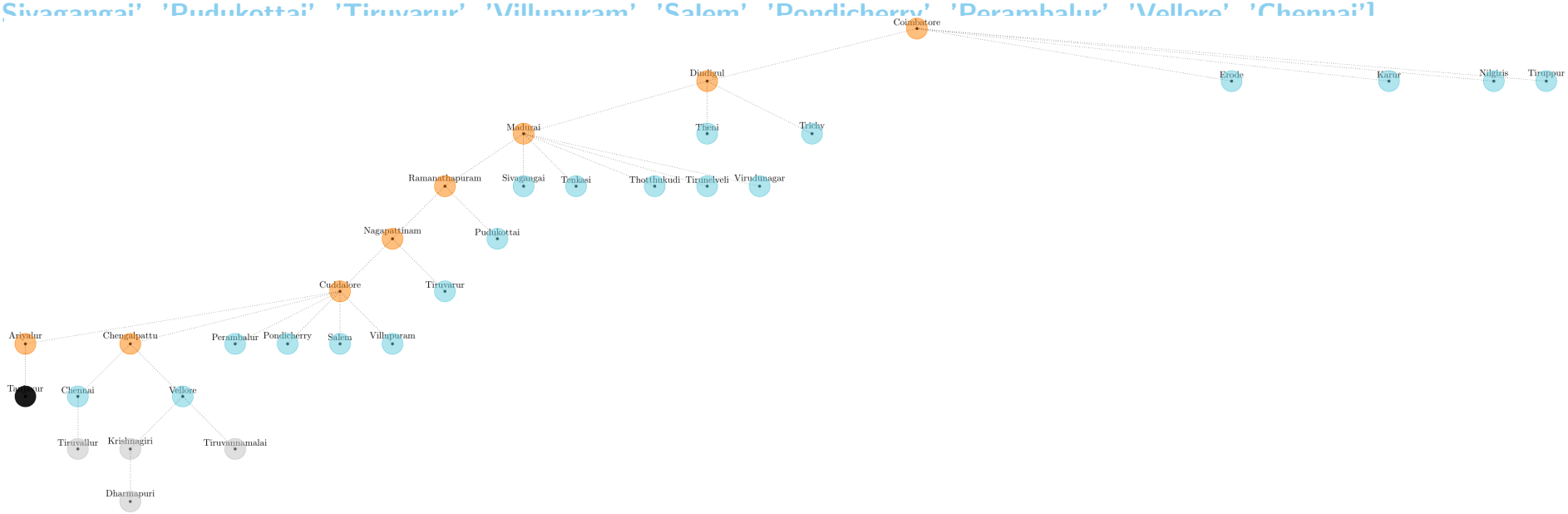
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Chengalpattu']

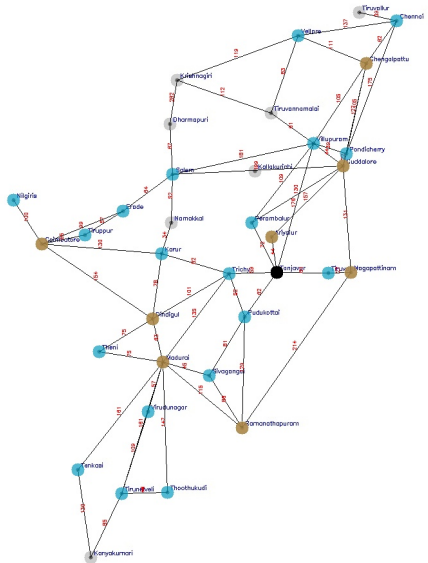




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

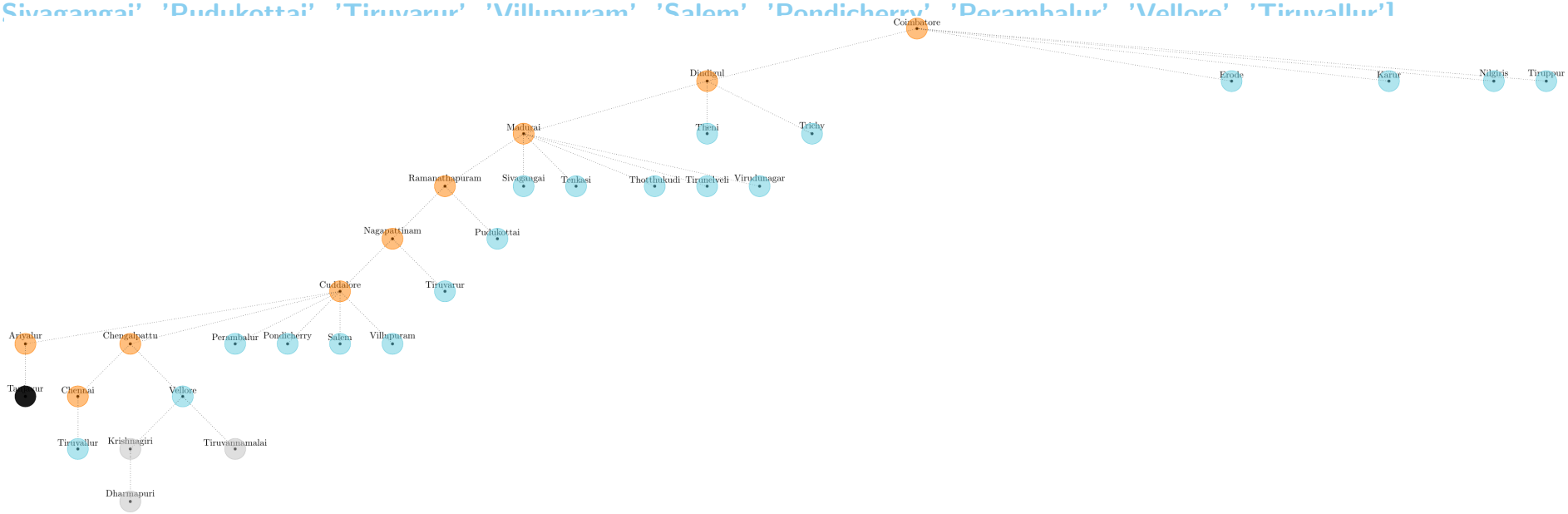
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Vellore', 'Chennai']

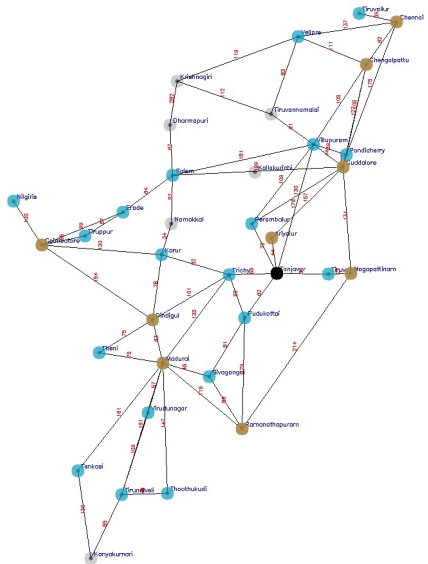




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

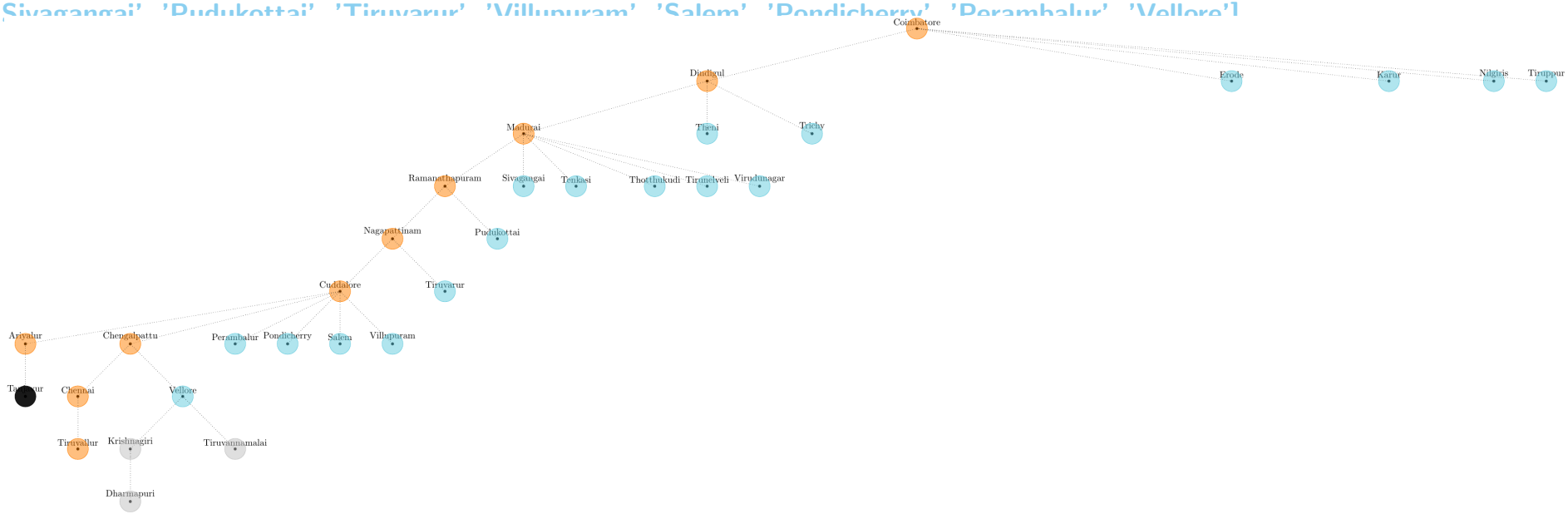
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Vellore', 'Tiruvallur']

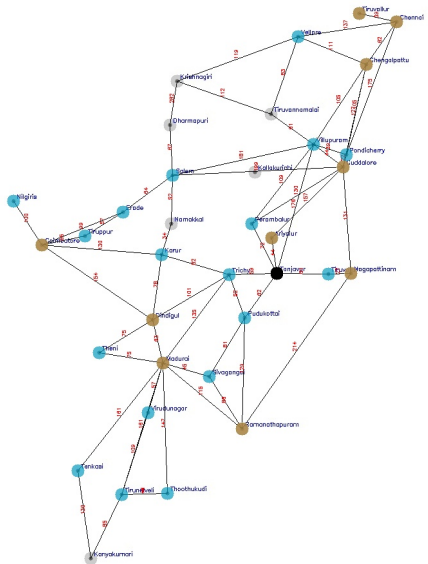




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

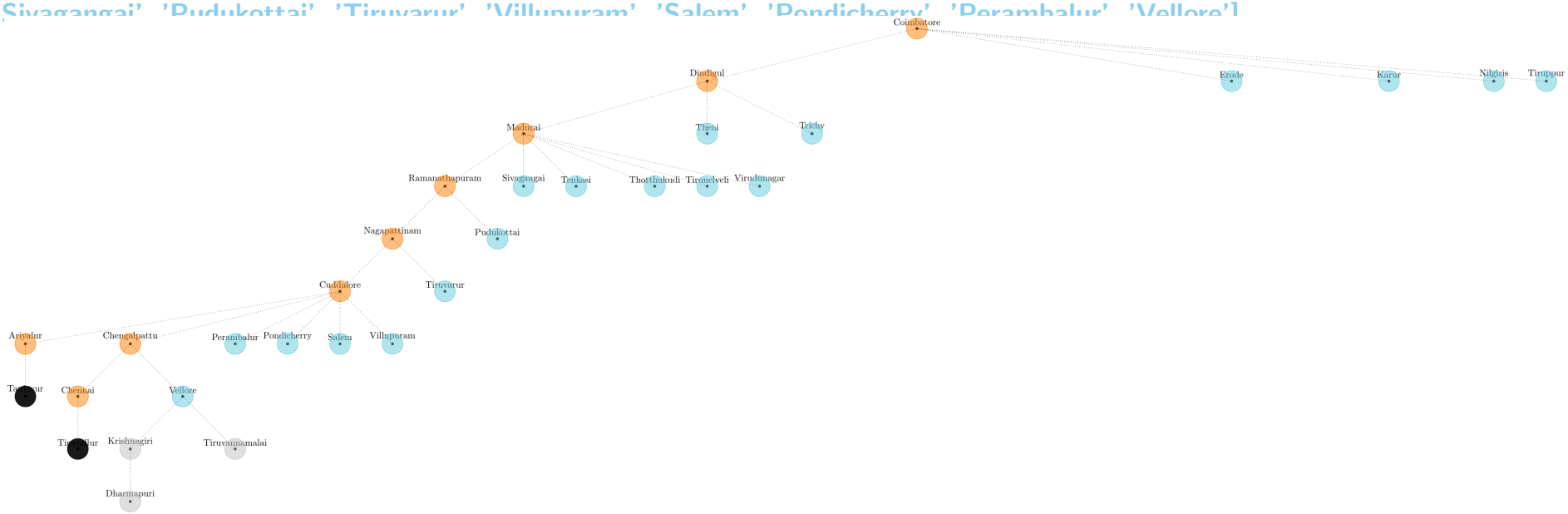
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Vellore']

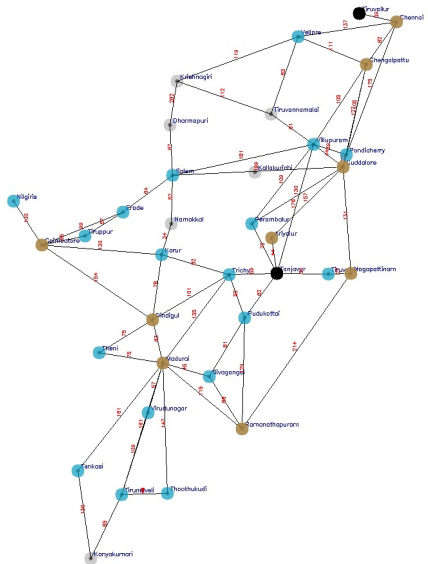




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

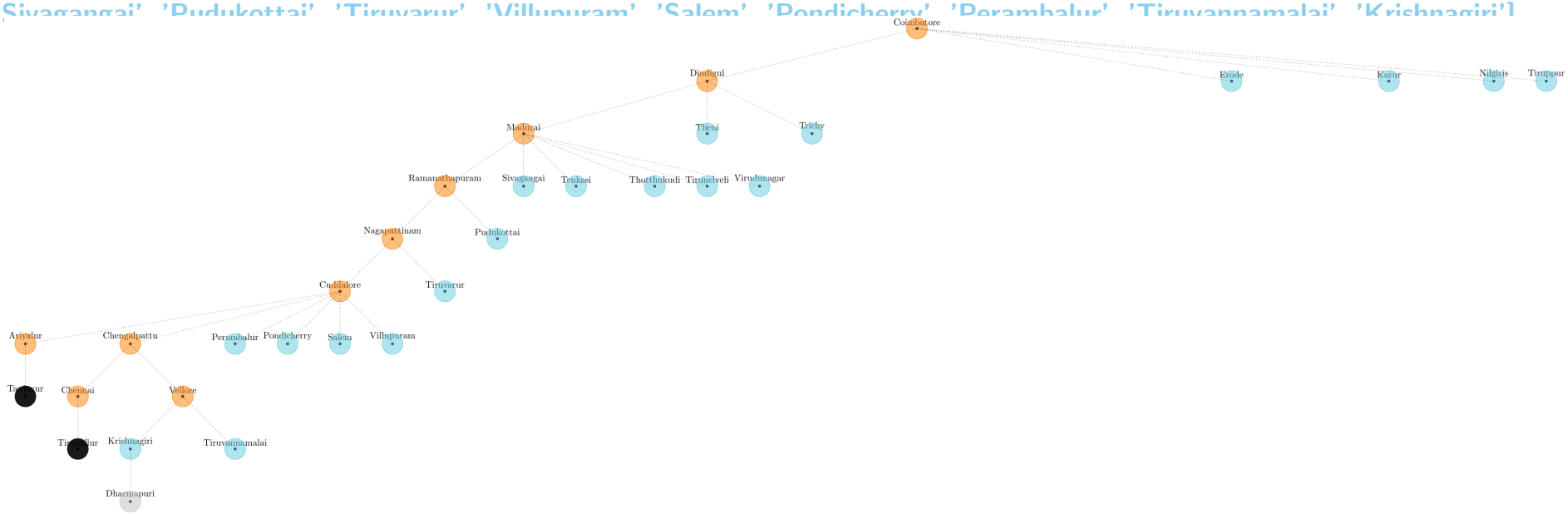
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Vellore']

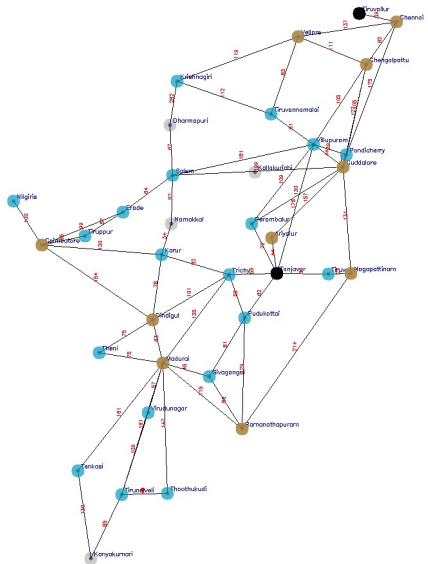




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

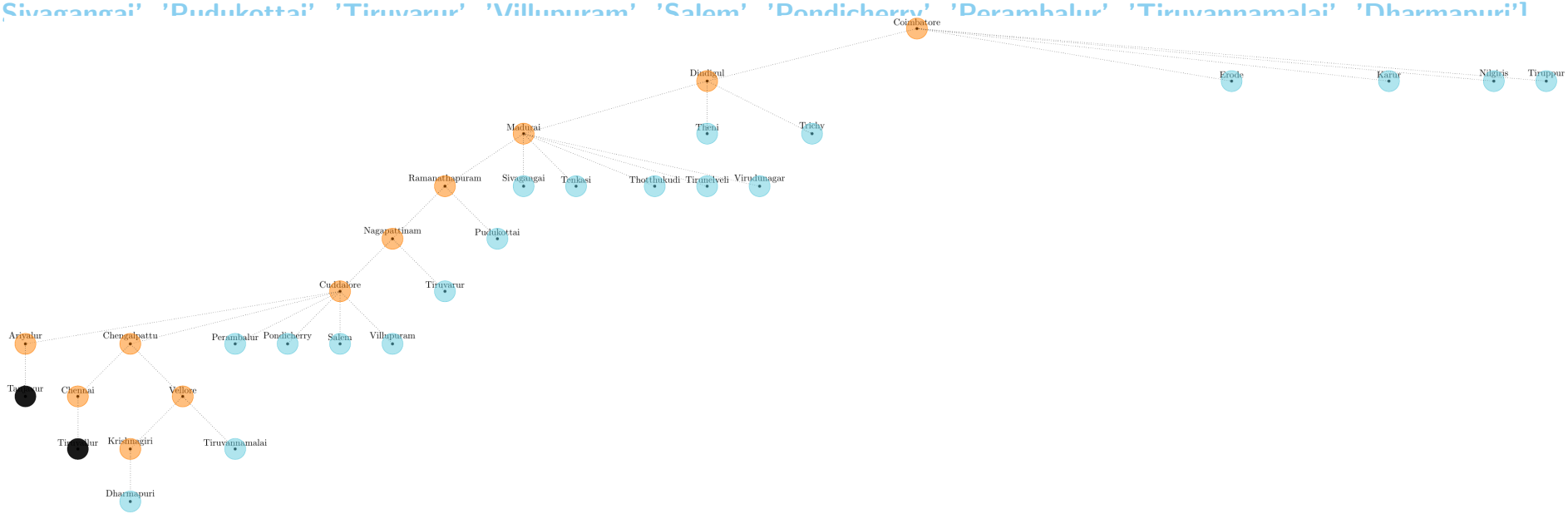
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Tiruvannamalai', 'Krishnagiri']

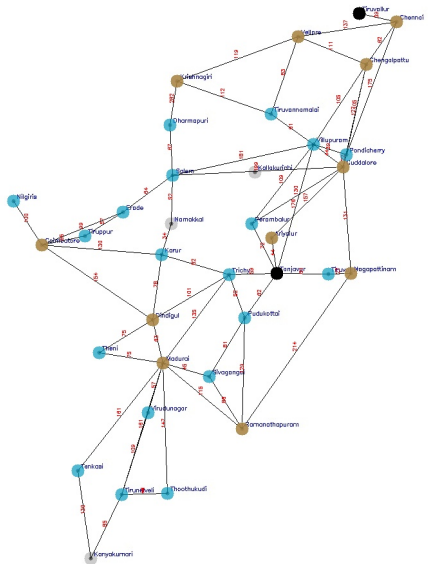




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

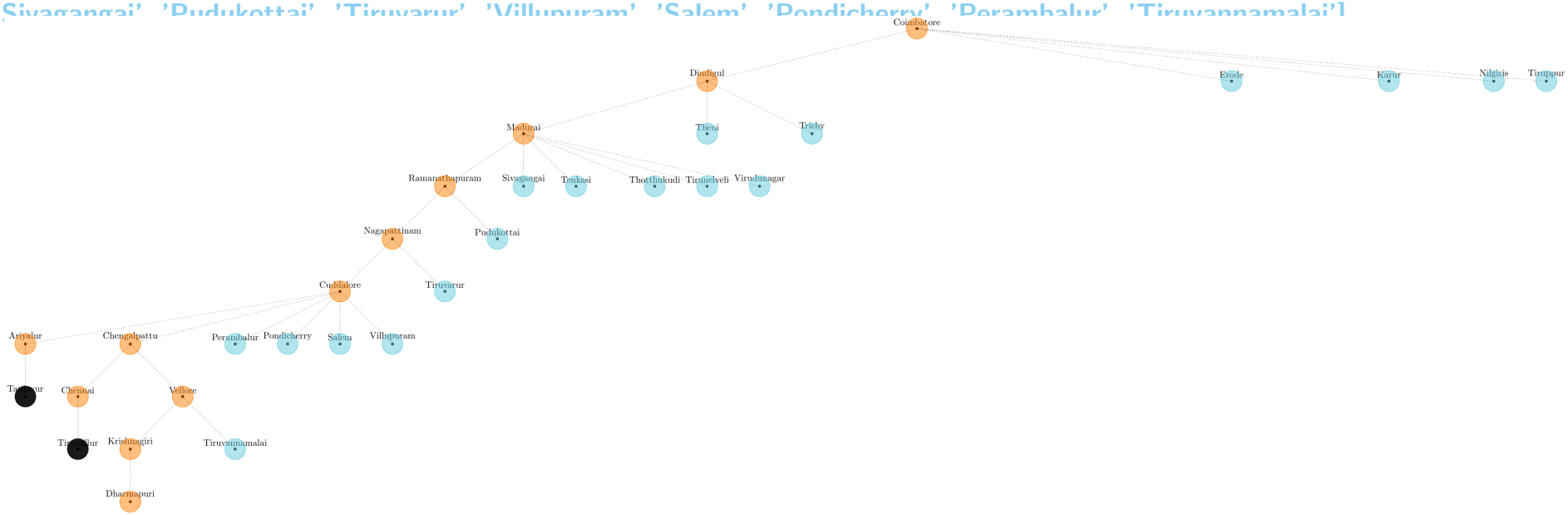
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Tiruvannamalai', 'Dharmapuri']

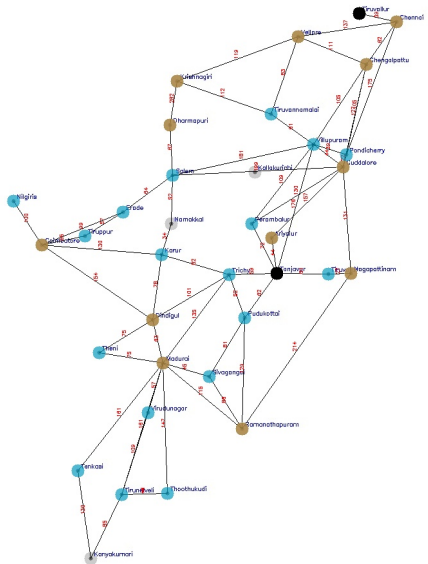




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

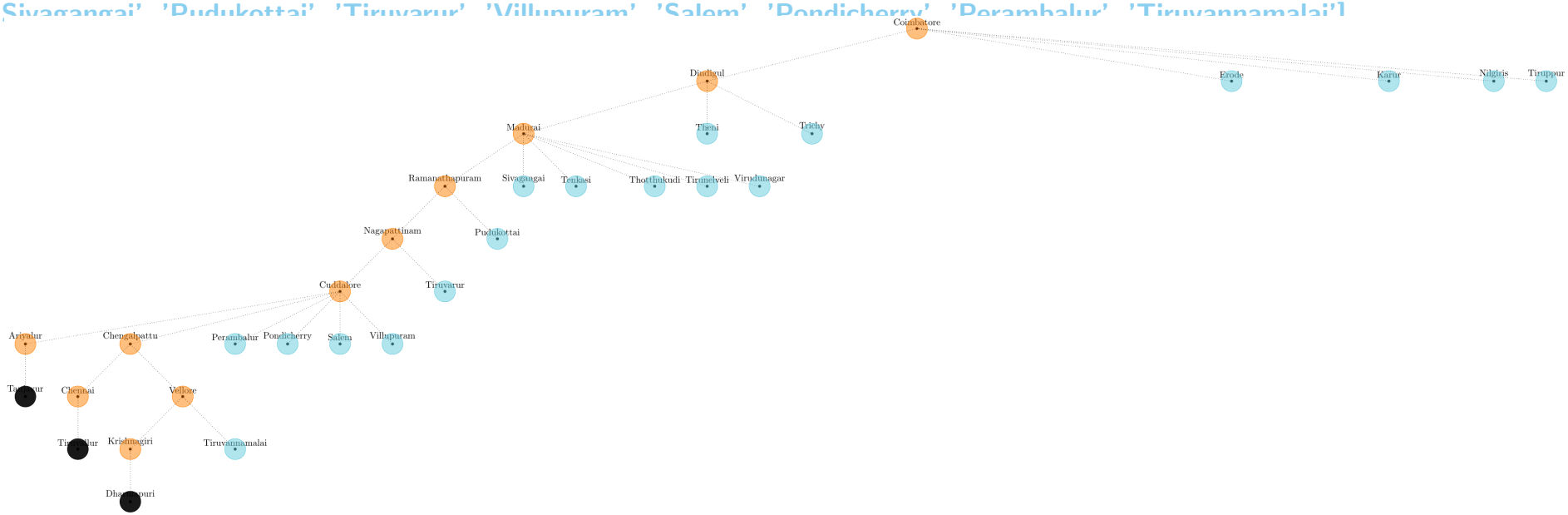
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Tiruvannamalai']

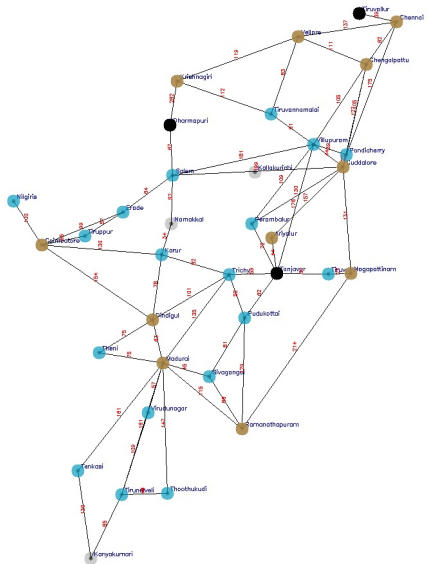




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

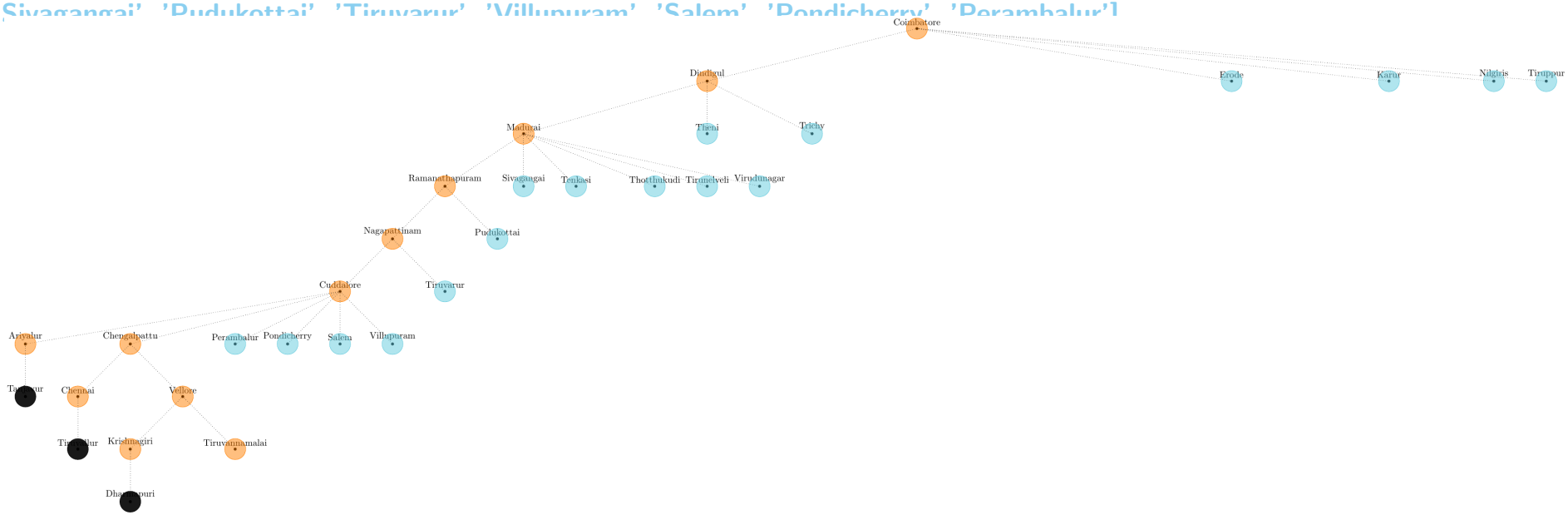
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur', 'Tiruvannamalai']

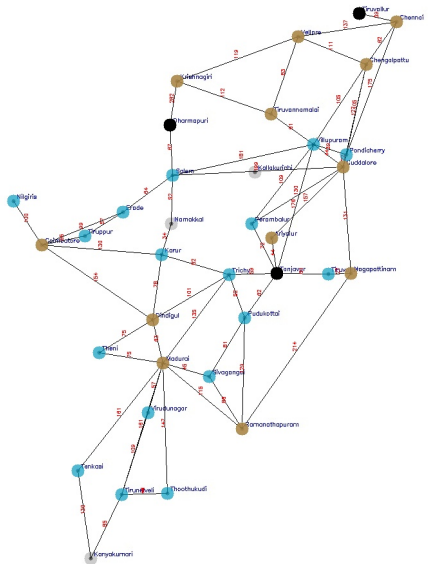




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

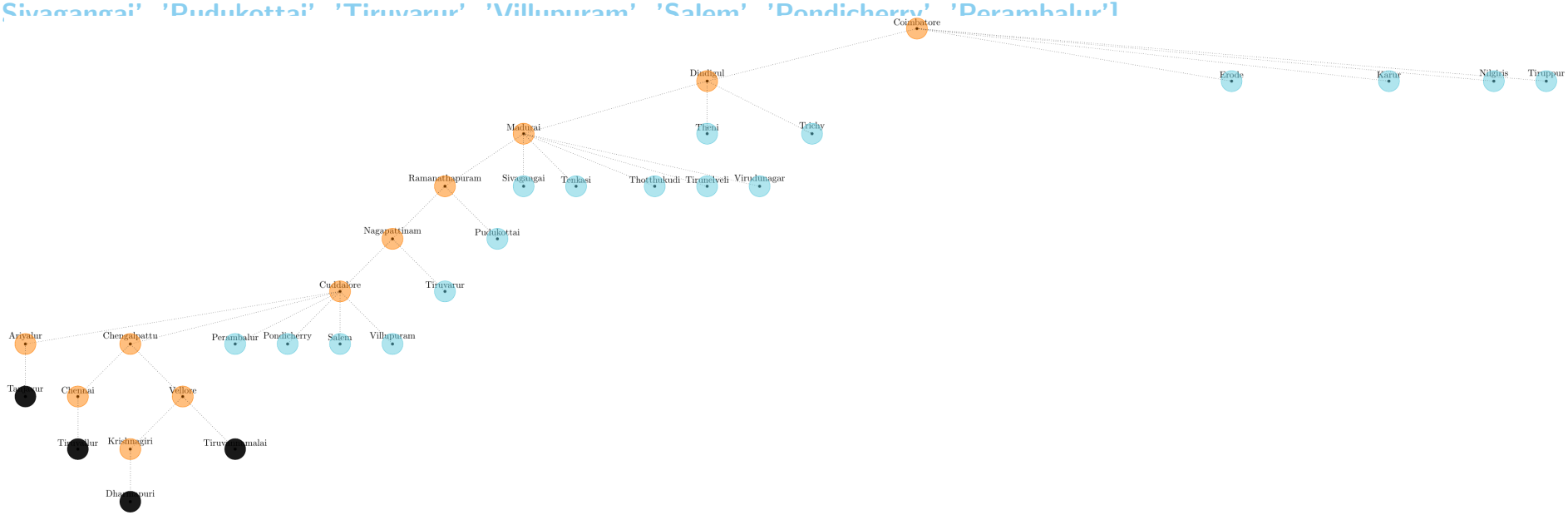
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur']

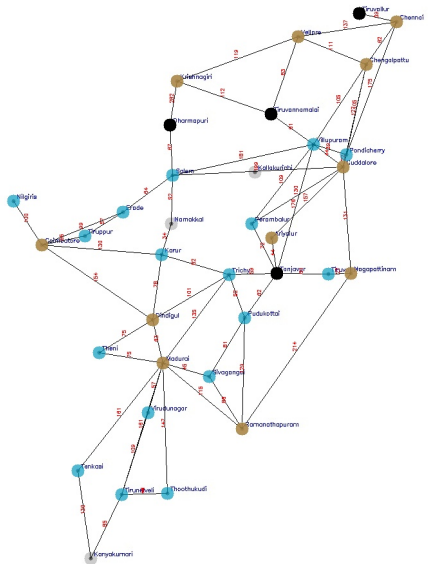




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

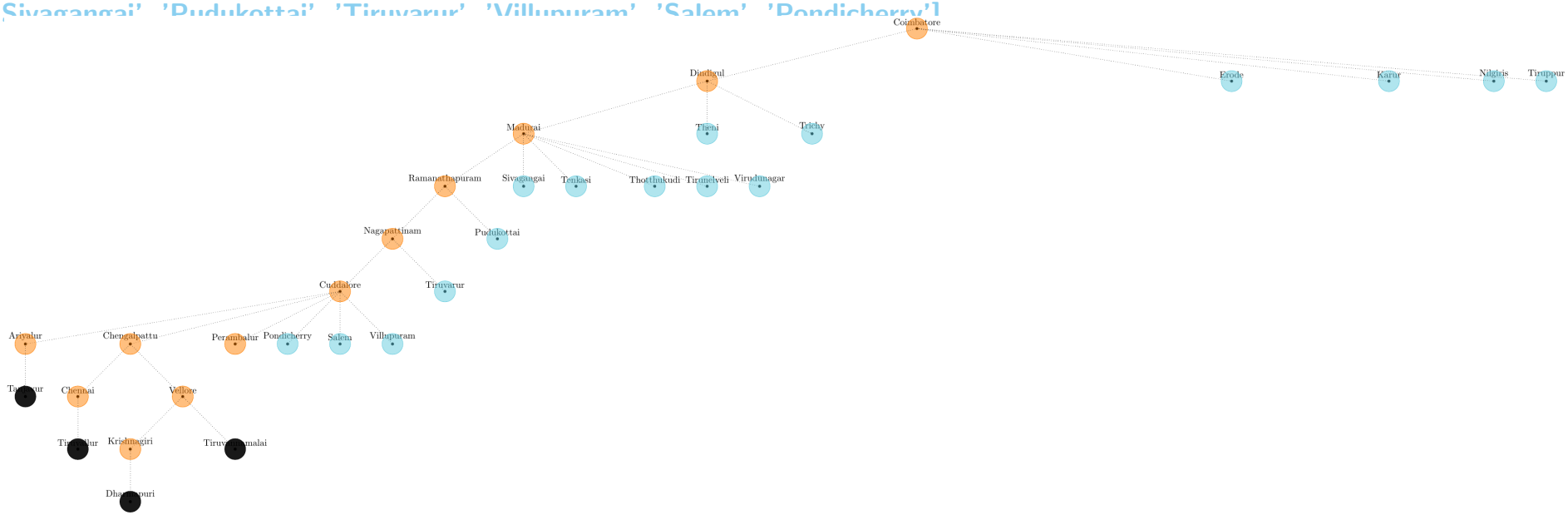
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry', 'Perambalur']

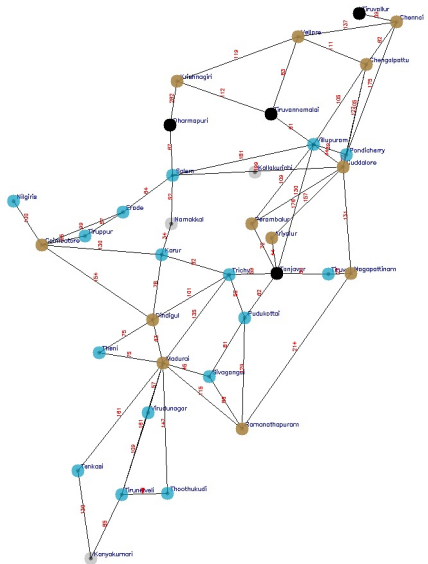




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

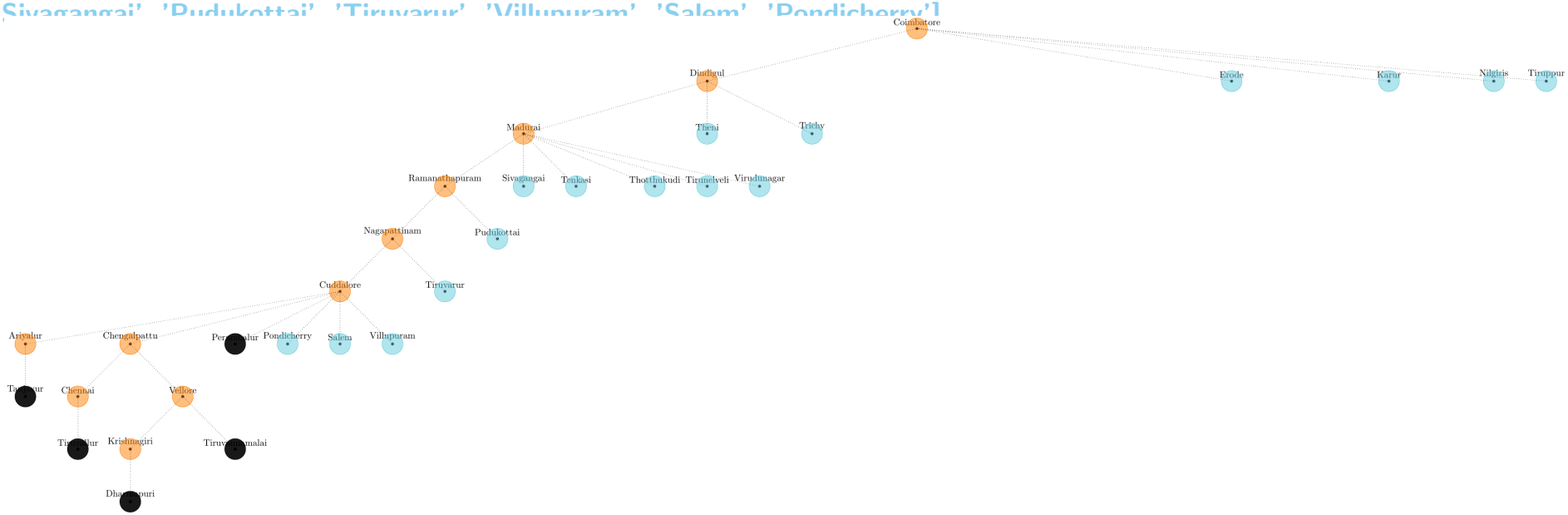
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry']

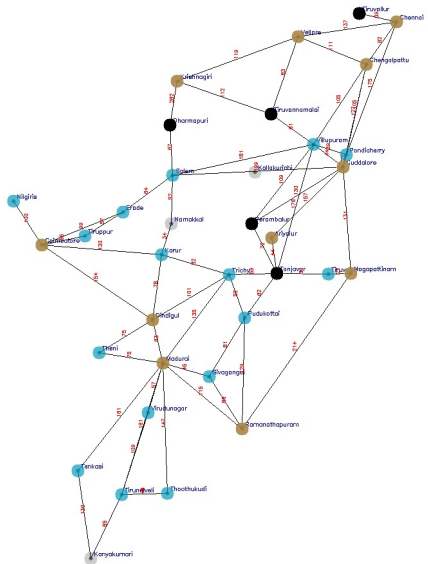




Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

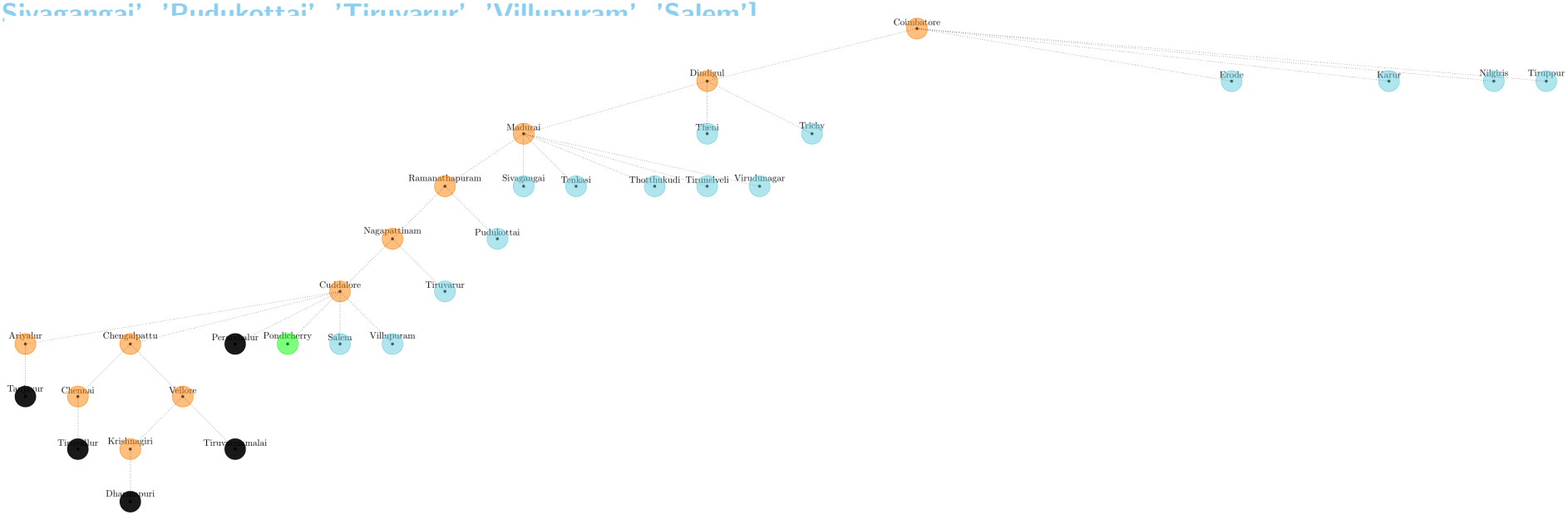
'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem', 'Pondicherry']

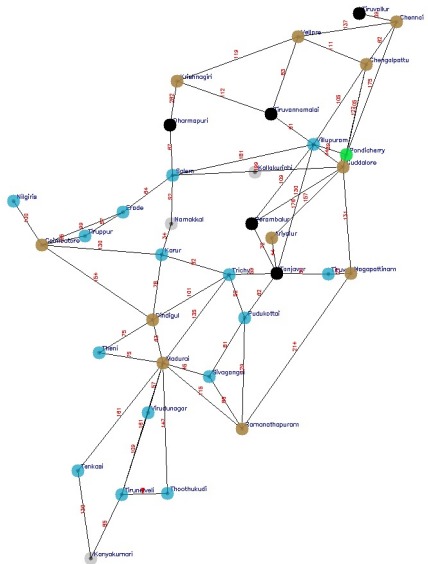




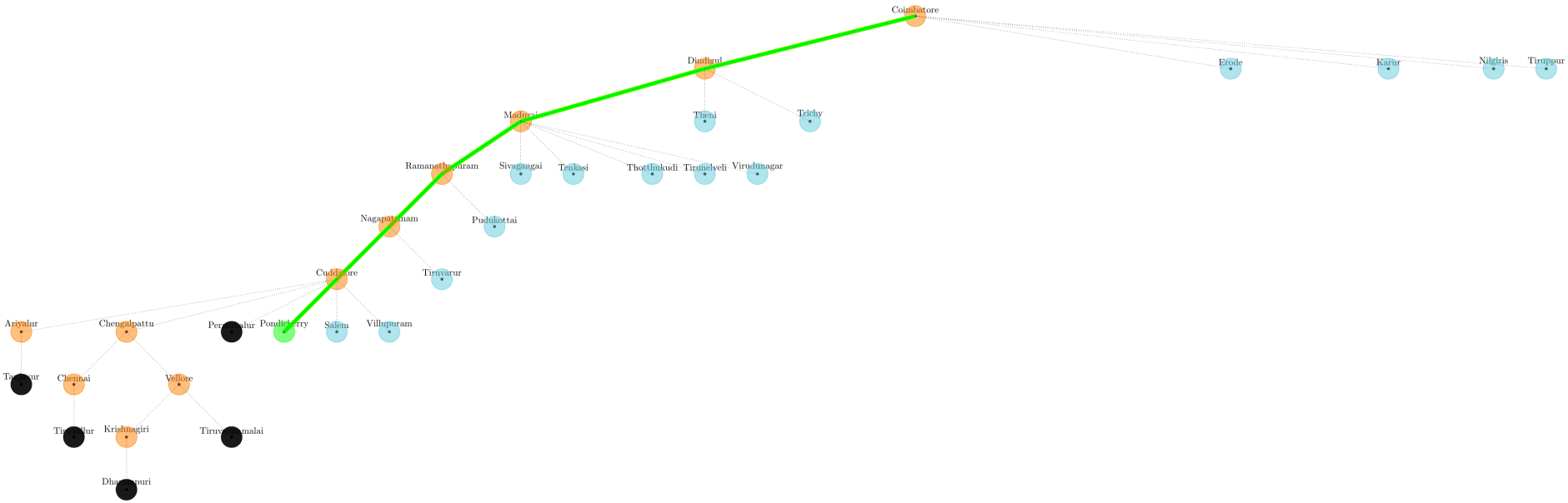
Fontier: ['Tiruppur', 'Nilgiris', 'Karur', 'Erode', 'Trichy', 'Theni', 'Virudunagar', 'Tirunelveli', 'Thoothukudi', 'Tenkasi',

'Sivagangai', 'Pudukottai', 'Tiruvarur', 'Villupuram', 'Salem']





The path DFS took is from Coimbatore to Pondicherry is Coimbatore $->$ Dindigul $->$ Madurai $->$ Ramanathapuram $->$ Nagapattinam $->$ Cuddalore $->$ Pondicherry.

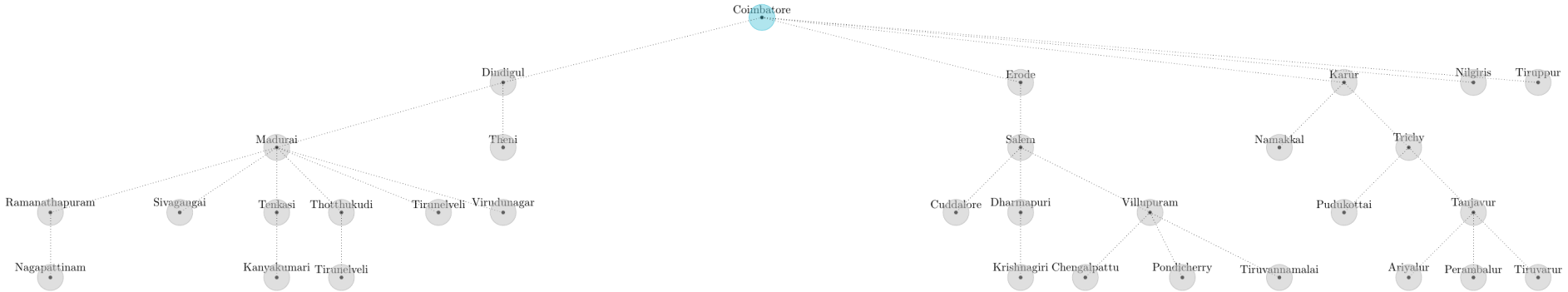


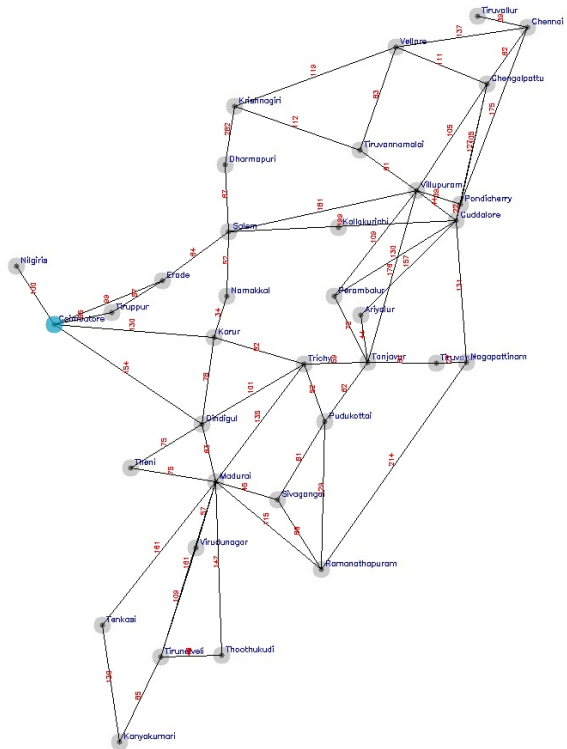
Unifrom Cost Search - UCS

- In UCS, we expand the least cost node.
- It doesn't use any domain information.
- The data structure used is a **HEAP**, we use the least cost as function.

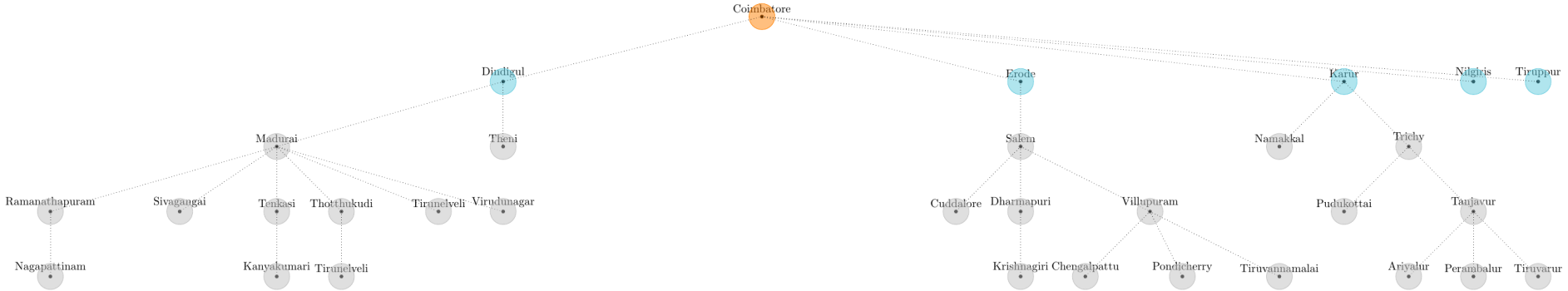
```
function UNIFORM-COST-SEARCH(initialState, goalTest)
  returns SUCCESS or FAILURE: /* Cost f(n) = g(n) */

  frontier = Heap.new(initialState)
  explored = Set.new()

  while not frontier.isEmpty()
    state = frontier.deleteMin()
    explored.add(state)
    if goalTest(state):
      return SUCCESS(state)

    for neighbor in state.neighbors():
      if neighbor not in frontier U explored:
        frontier.insert(neighbor)
      else if neighbor in frontier:
        frontier.decreaseKey(neighbor)
  return FAILURE
```

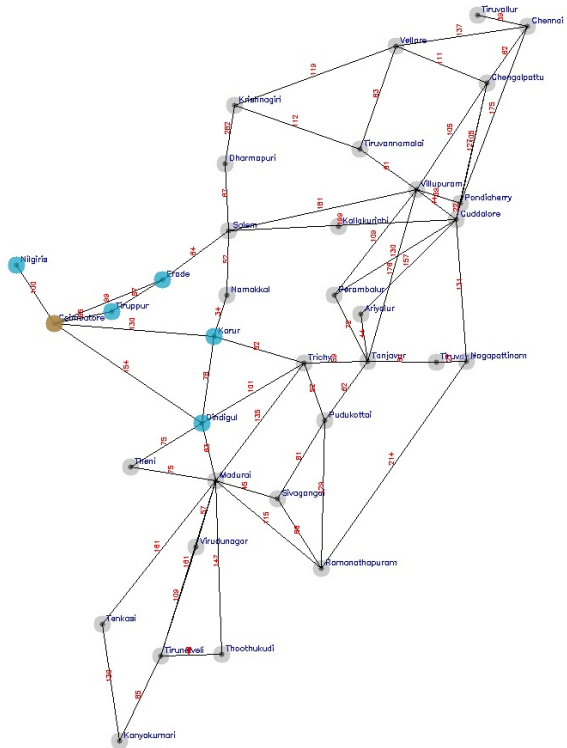
Fontier: ['Coimbatore - 0']



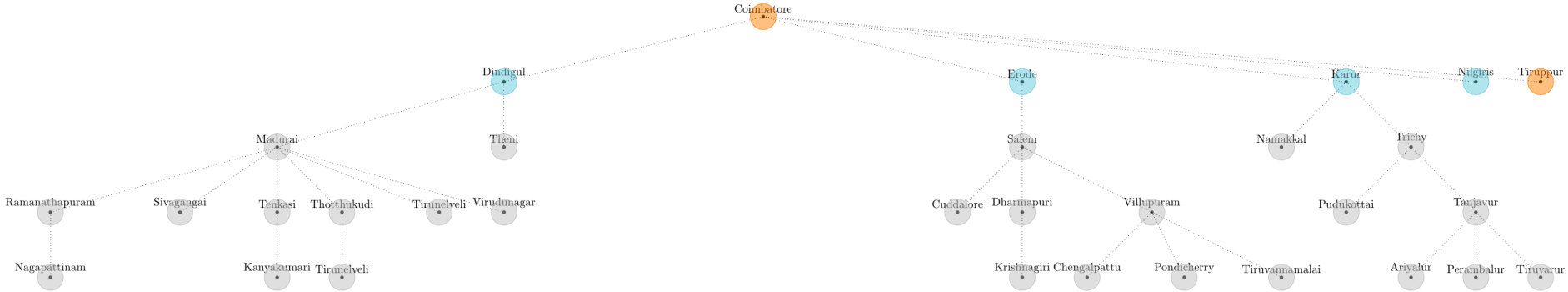


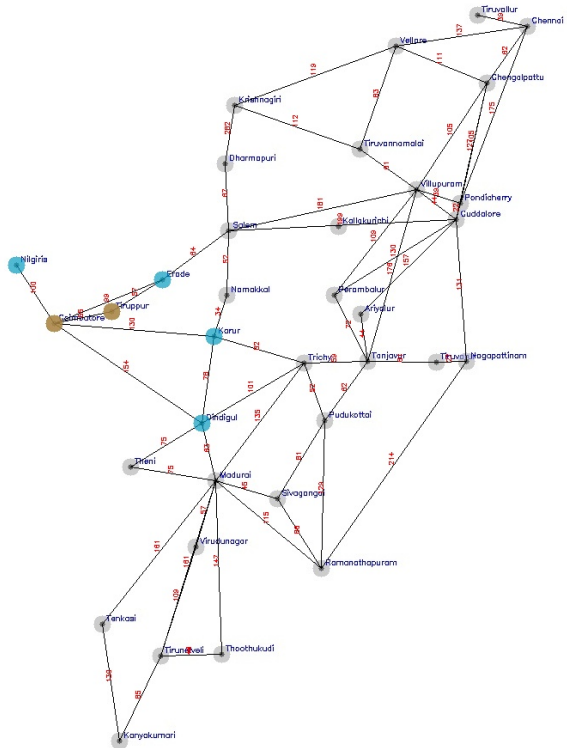
Fontier: ['Dindigul' - 154,'Erode' - 99,'Karur' - 130,'Nilgiris' - 100,'Tiruppur' - 55,]



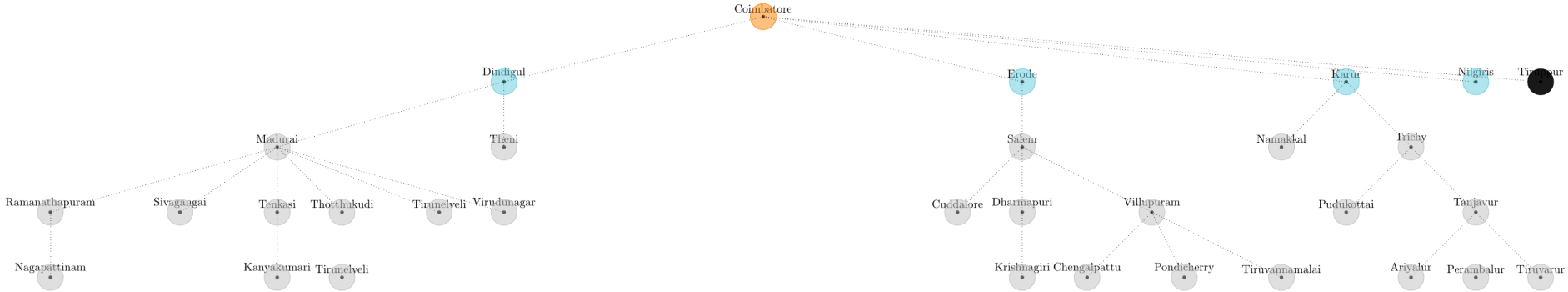


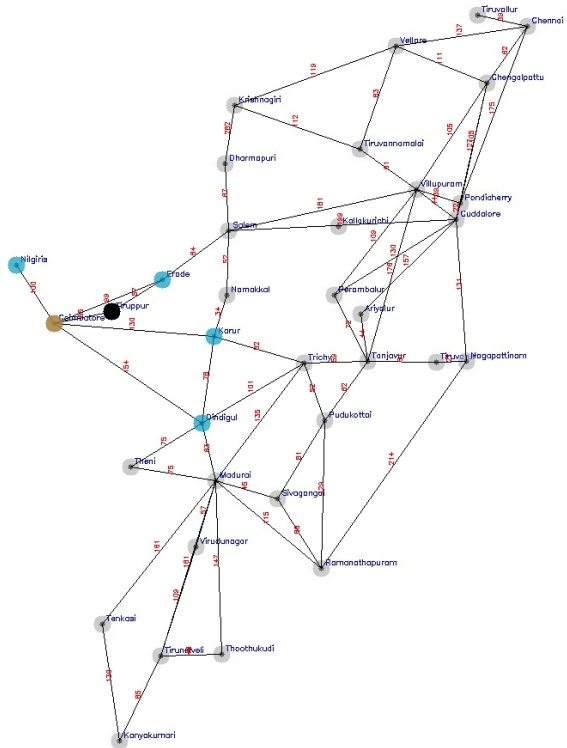
Fontier: ['Dindigul' - 154,'Erode' - 99,'Karur' - 130,'Nilgiris' - 100,]



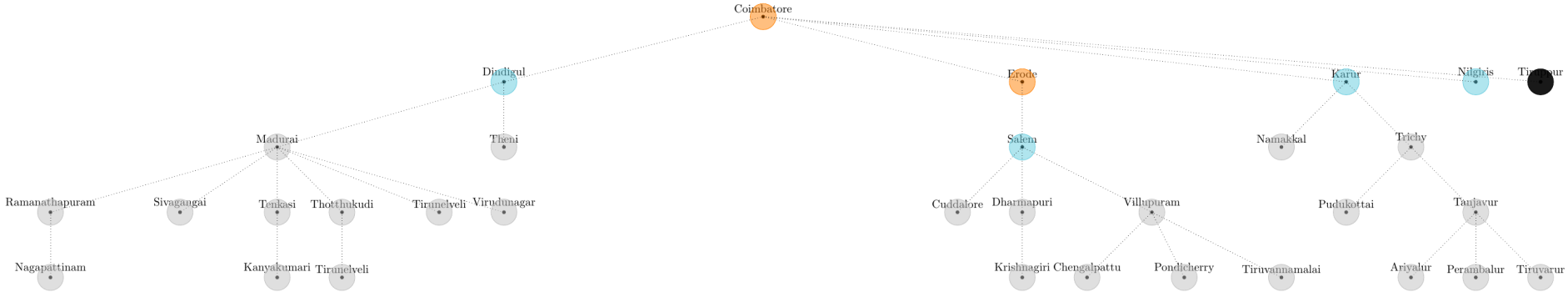


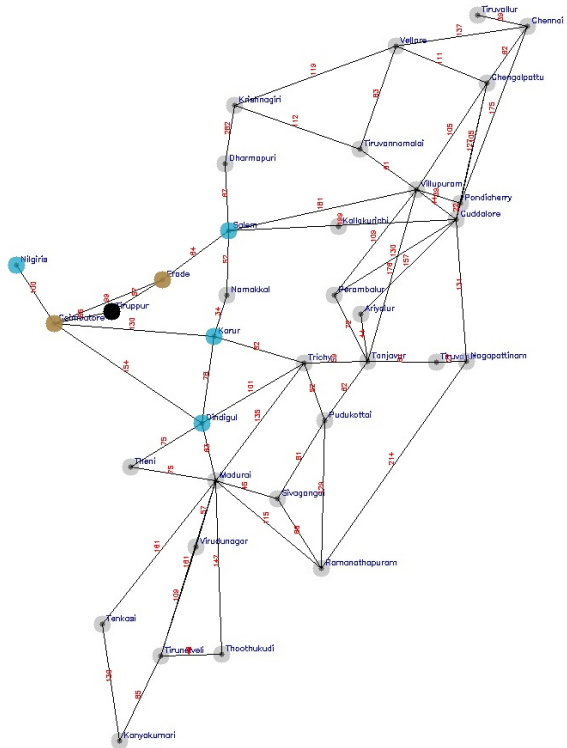
Fontier: ['Dindigul' - 154,'Erode' - 99,'Karur' - 130,'Nilgiris' - 100,]



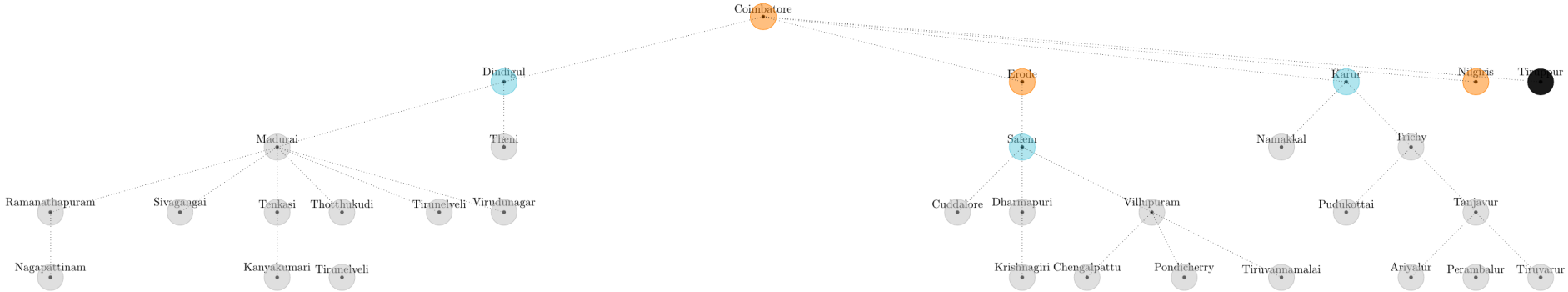


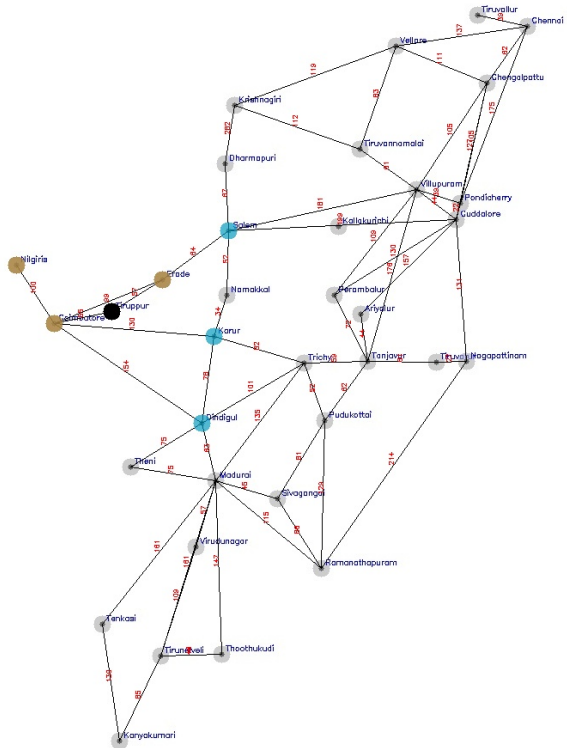
Fontier: ['Dindigul' - 154,'Karur' - 130,'Nilgiris' - 100,'Salem' - 163,]



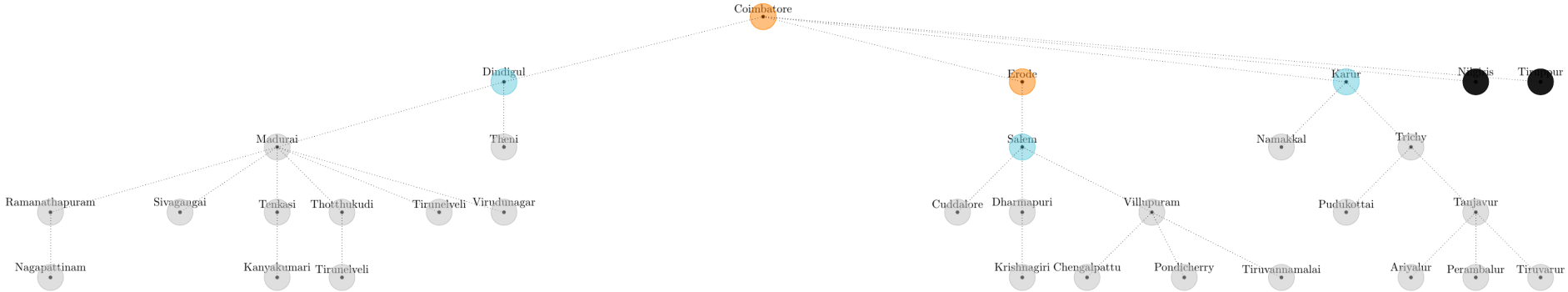


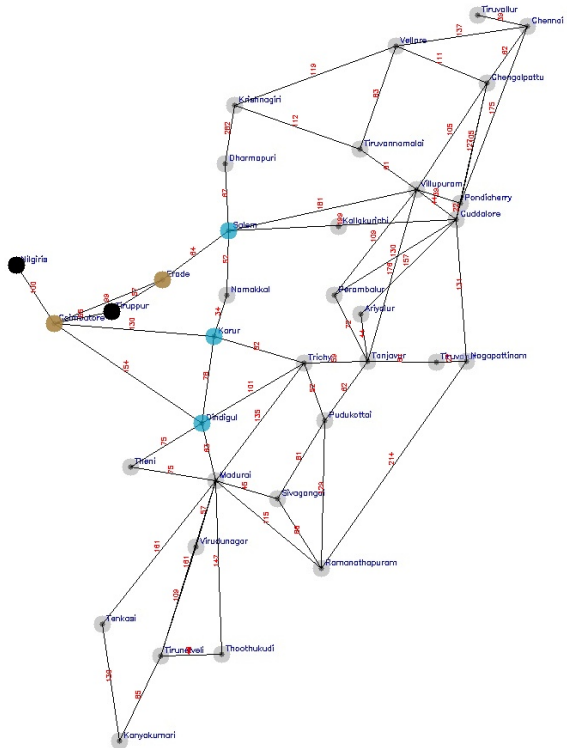
Fontier: ['Dindigul' - 154,'Karur' - 130,'Salem' - 163,]



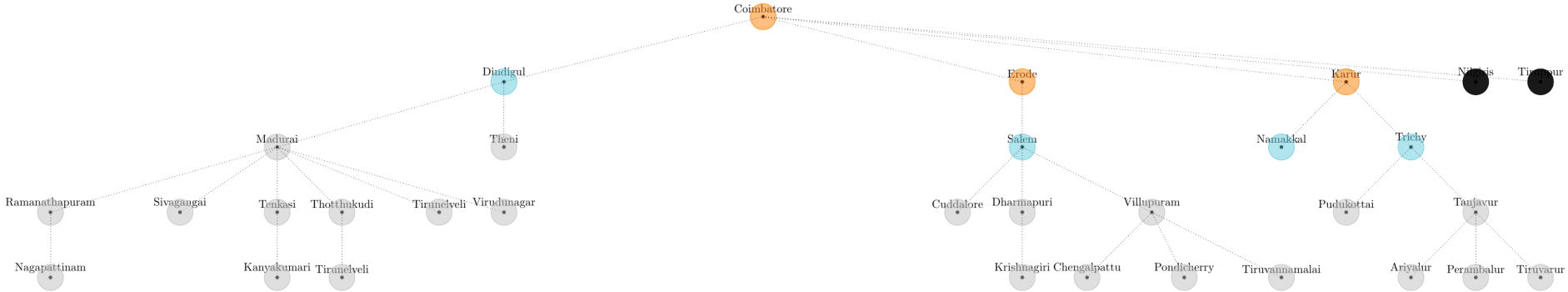


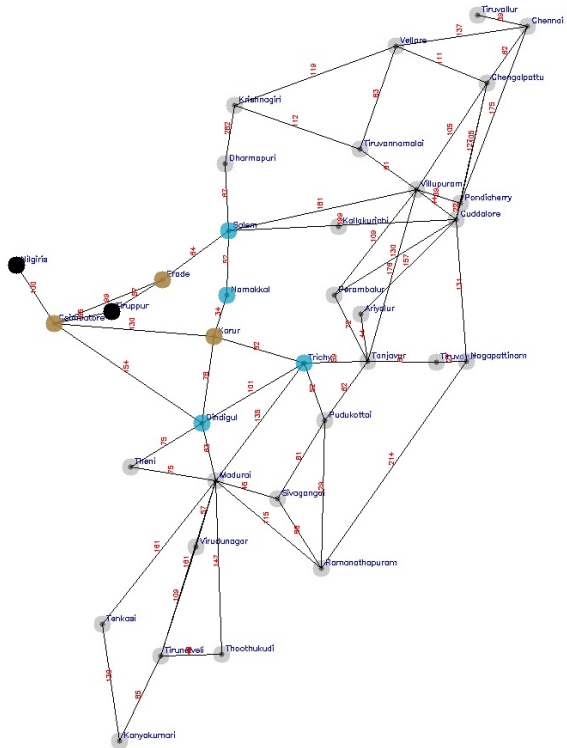
Fontier: ['Dindigul' - 154,'Karur' - 130,'Salem' - 163,]



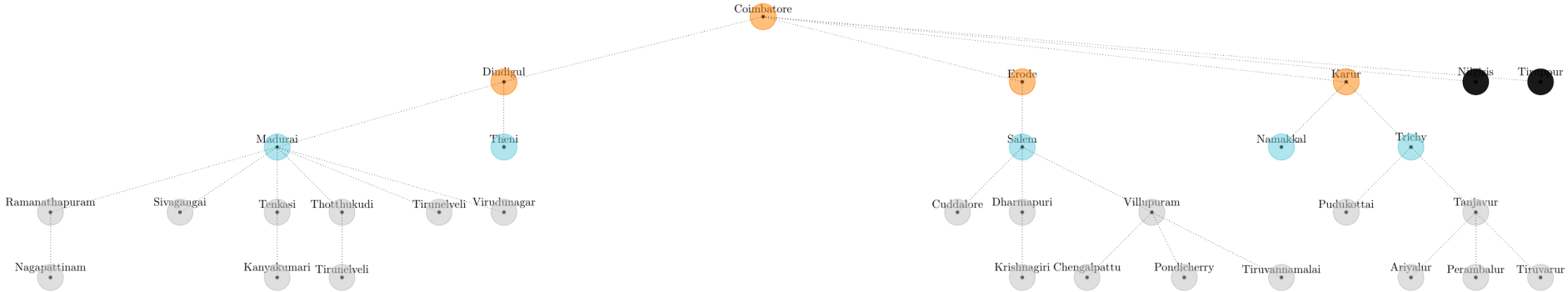


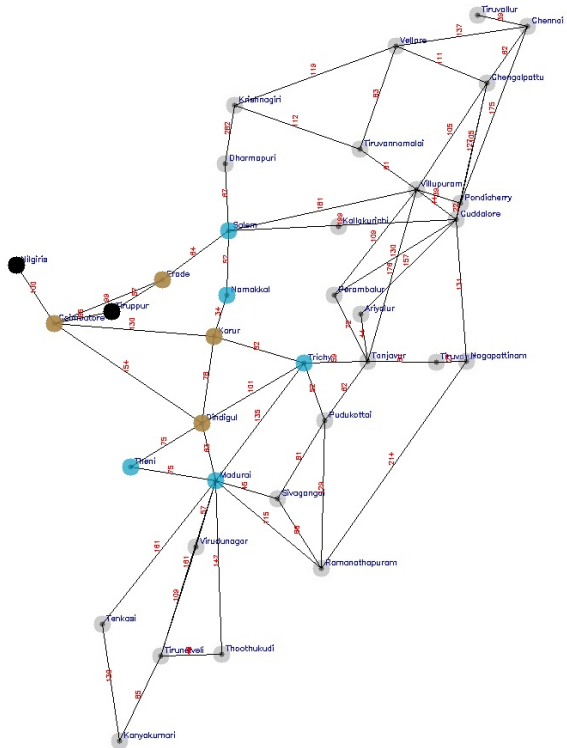
Fontier: ['Dindigul' - 154,'Salem' - 163,'Namakkal' - 164,'Trichy' - 212,]



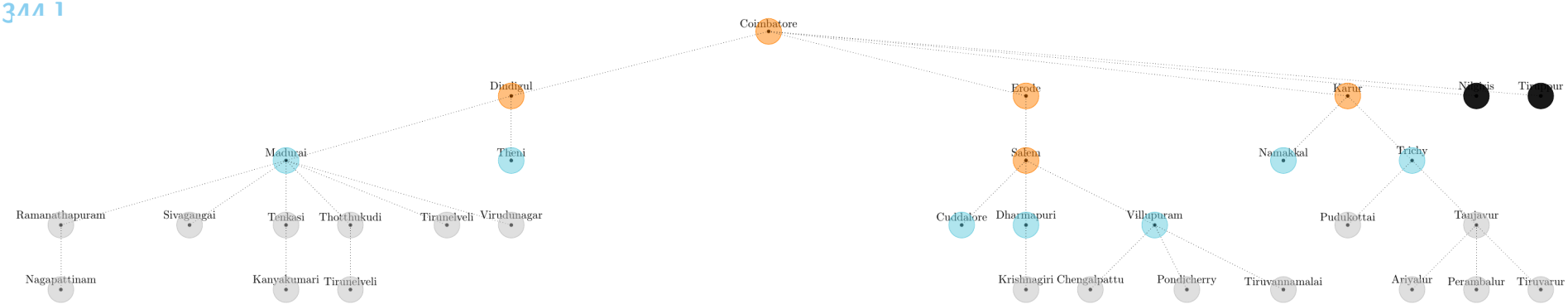


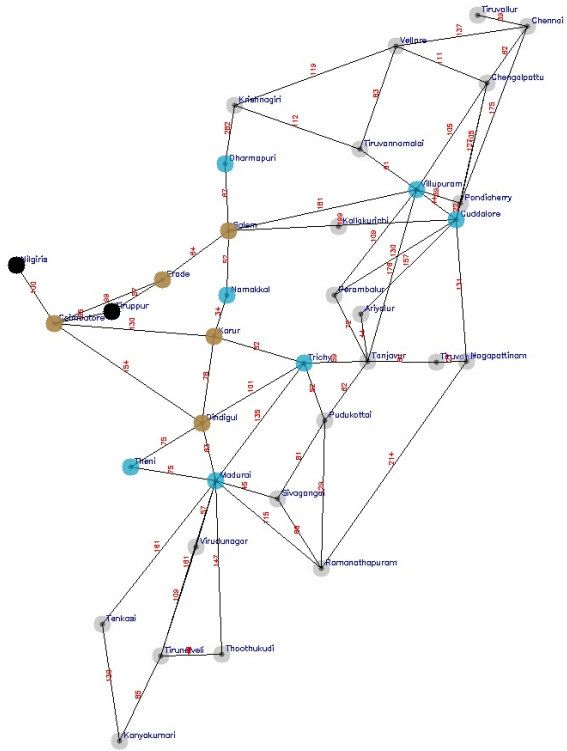
Fontier: ['Salem' - 163,'Namakkal' - 164,'Trichy' - 212,'Madurai' - 217,'Theni' - 229,]



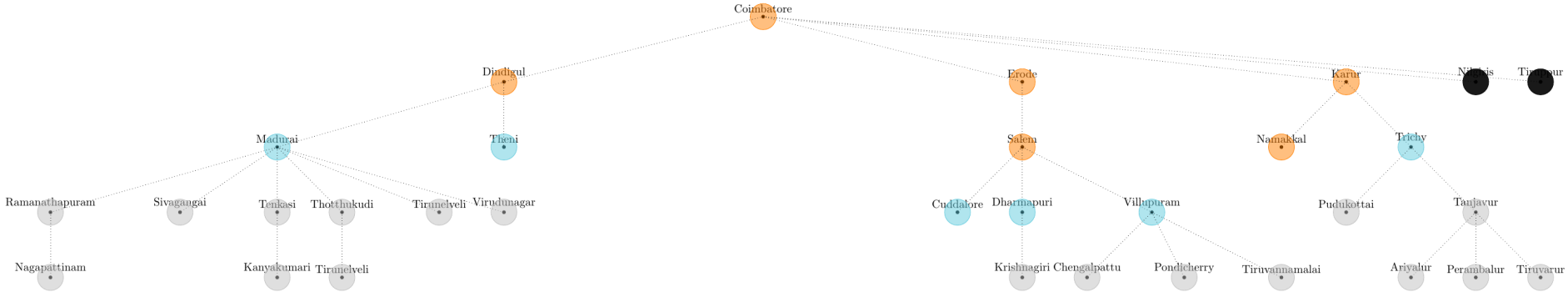


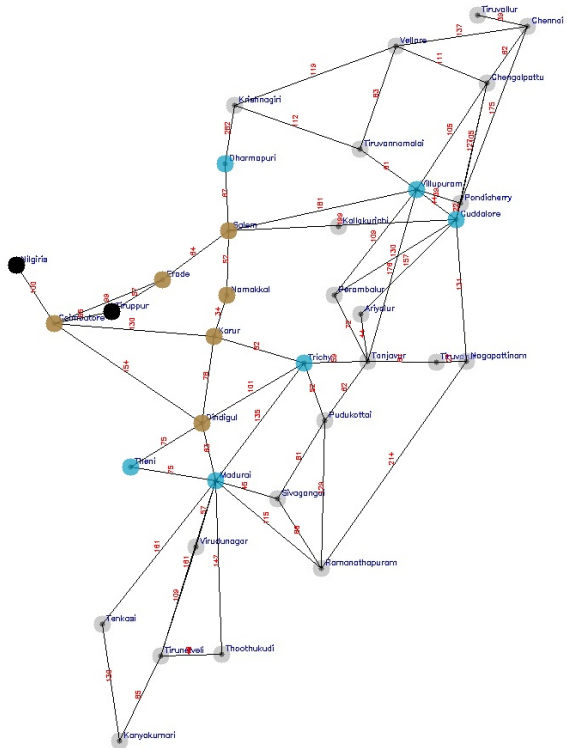
Fontier: ['Namakkal' - 164,'Trichy' - 212,'Madurai' - 217,'Theni' - 229,'Cuddalore' - 362,'Dharmapuri' - 230,'Villupuram' - 344]



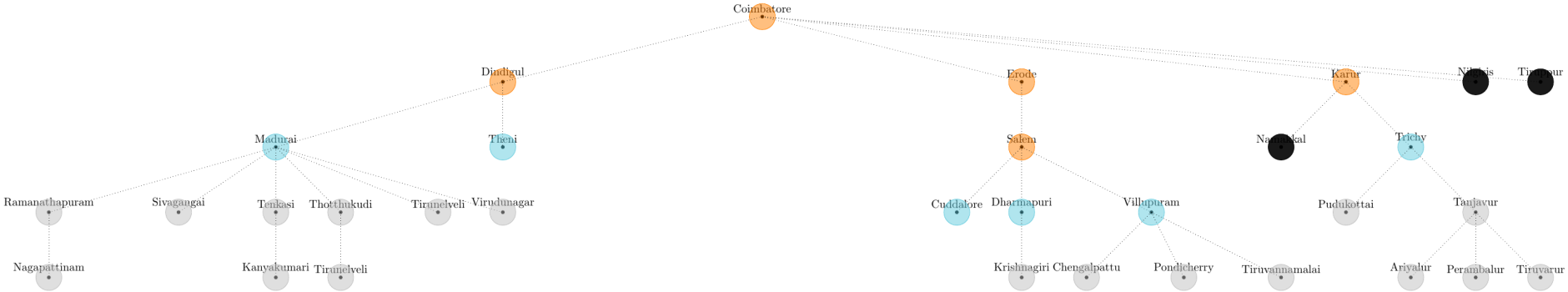


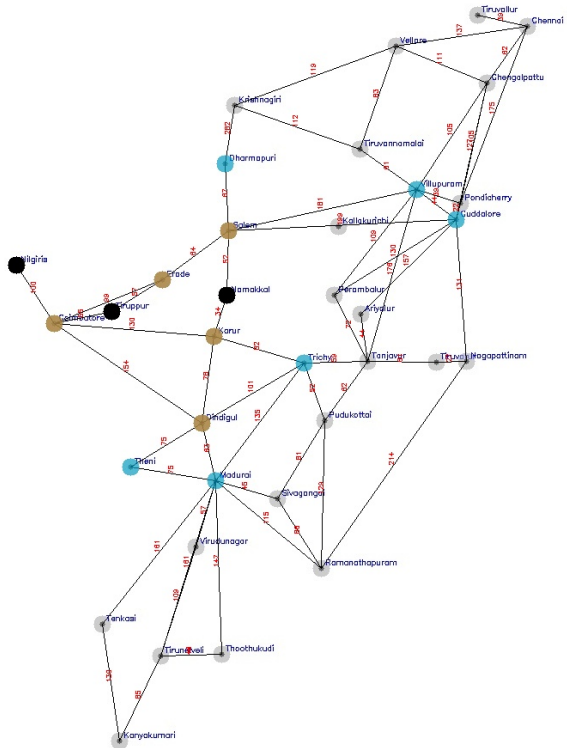
Fontier: ['Trichy' - 212,'Madurai' - 217,'Theni' - 229,'Cuddalore' - 362,'Dharmapuri' - 230,'Villupuram' - 344,]



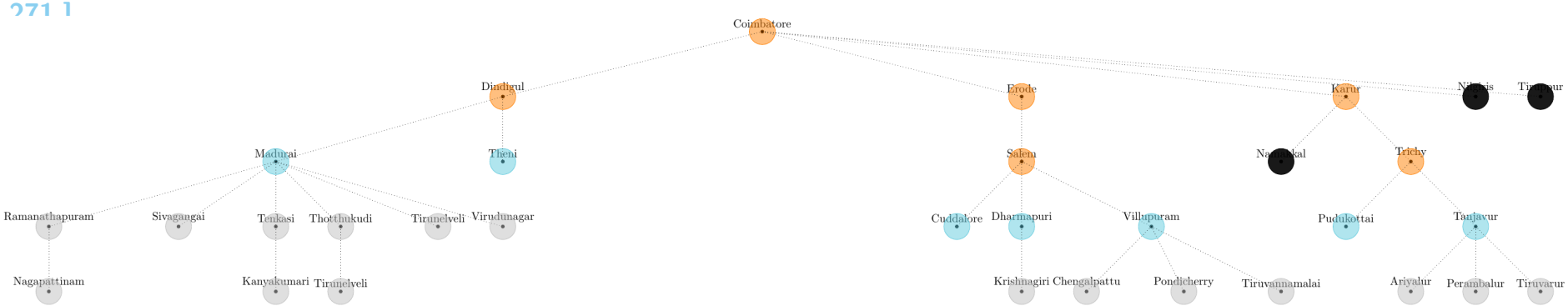


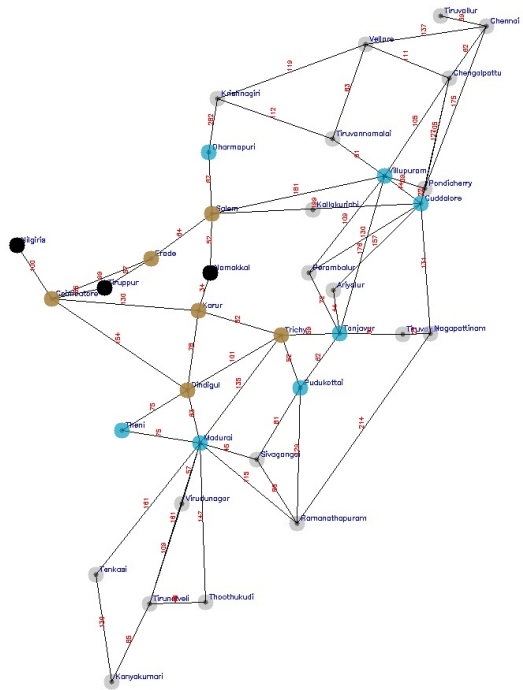
Fontier: ['Trichy' - 212,'Madurai' - 217,'Theni' - 229,'Cuddalore' - 362,'Dharmapuri' - 230,'Villupuram' - 344,]





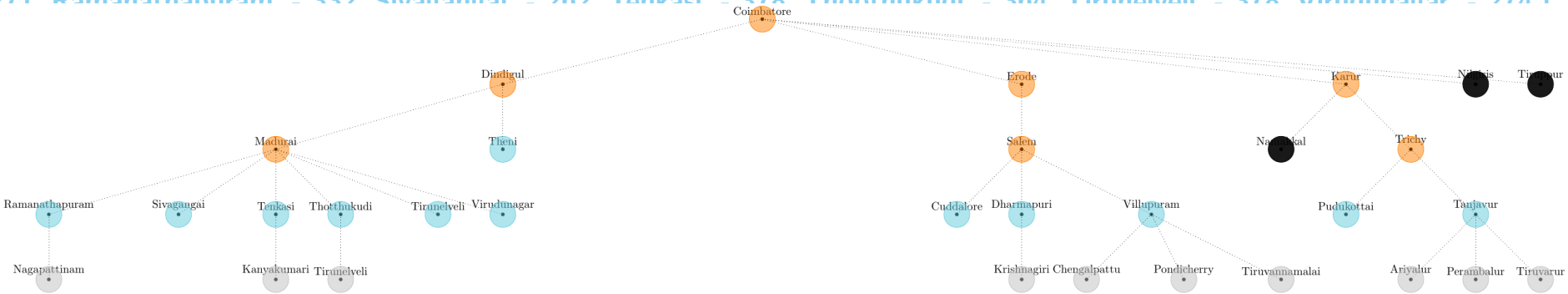
Fontier: ['Madurai' - 217,'Theni' - 229,'Cuddalore' - 362,'Dharmapuri' - 230,'Villupuram' - 344,'Pudukottai' - 264,'Tanjavur' - 271]

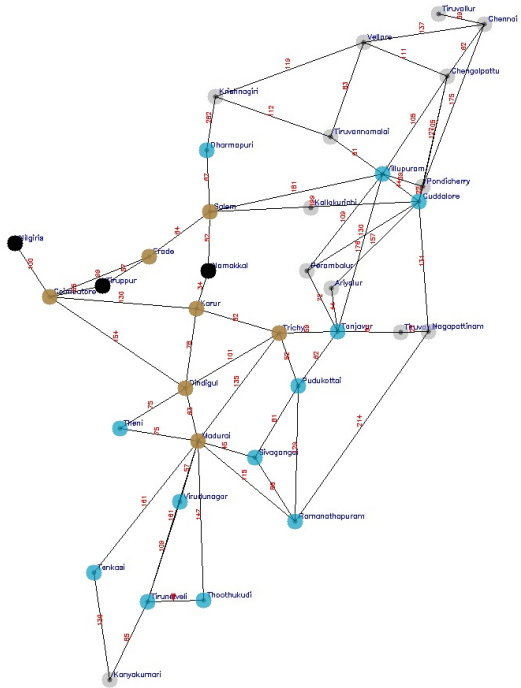




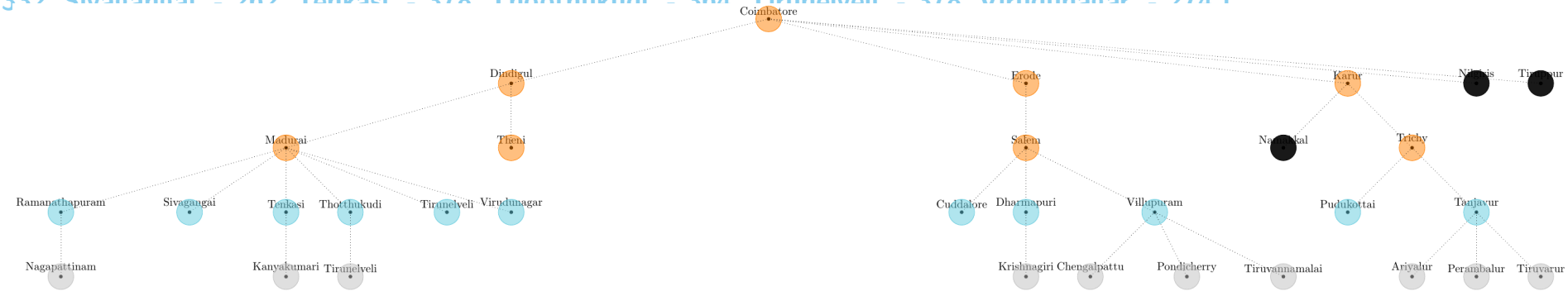
Fontier: ['Theni' - 229,'Cuddalore' - 362,'Dharmapuri' - 230,'Villupuram' - 344,'Pudukottai' - 264,'Tanjavur' -

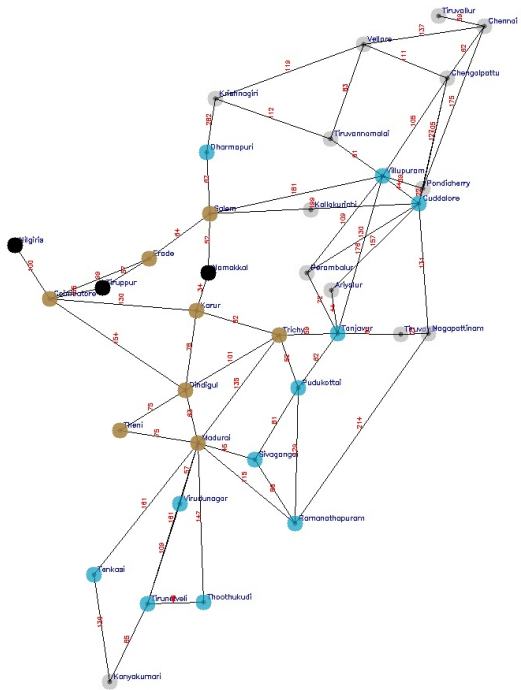
271,'Ramanathapuram' - 332,'Sivagangai' - 262,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Virudunagar' - 274]



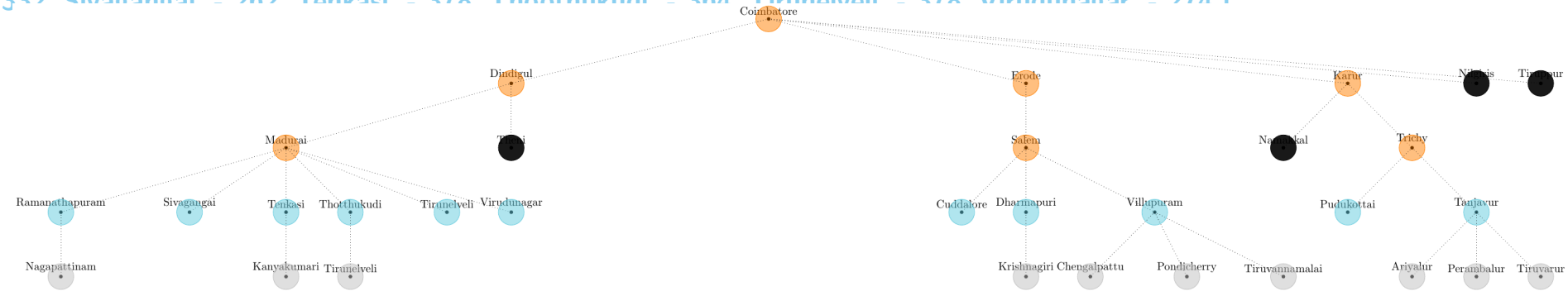


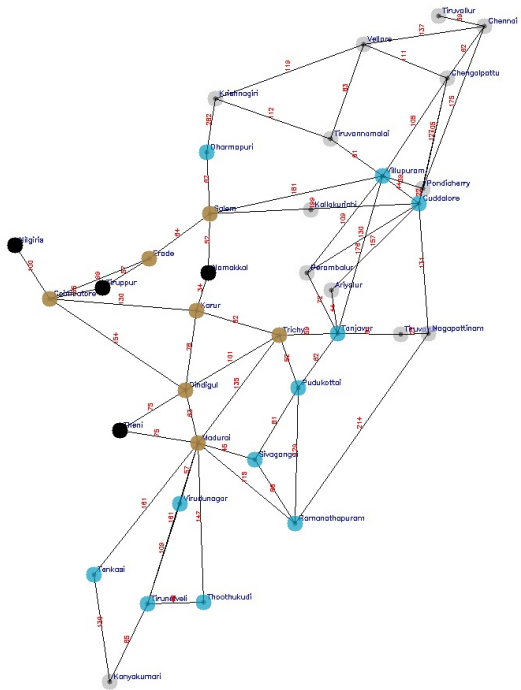
Fontier: ['Cuddalore' - 362,'Dharmapuri' - 230,'Villupuram' - 344,'Pudukottai' - 264,'Tanjavur' - 271,'Ramanathapuram' - 332,'Sivagangai' - 262,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Virudunagar' - 274,]



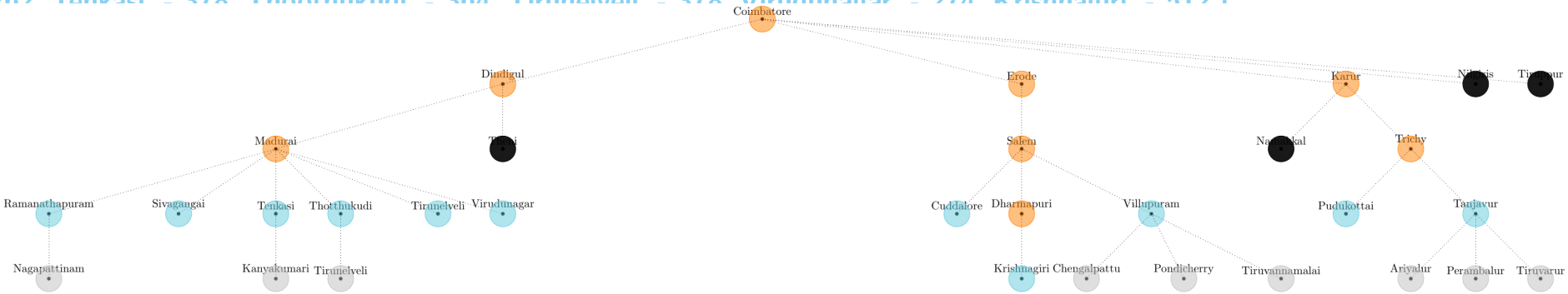


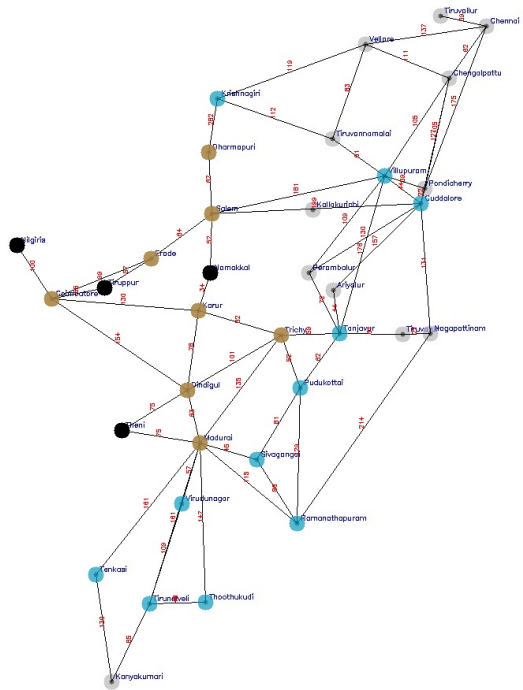
Fontier: ['Cuddalore' - 362,'Dharmapuri' - 230,'Villupuram' - 344,'Pudukottai' - 264,'Tanjavur' - 271,'Ramanathapuram' - 332,'Sivagangai' - 262,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Virudunagar' - 274,]



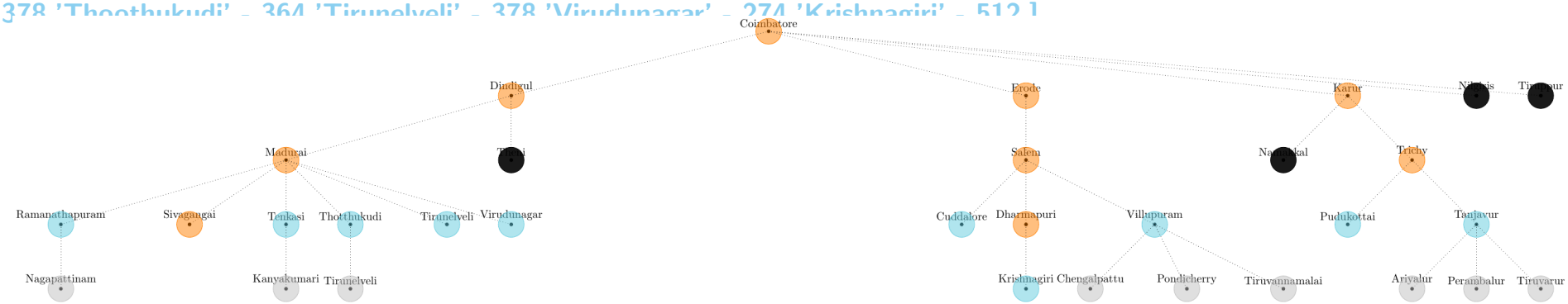


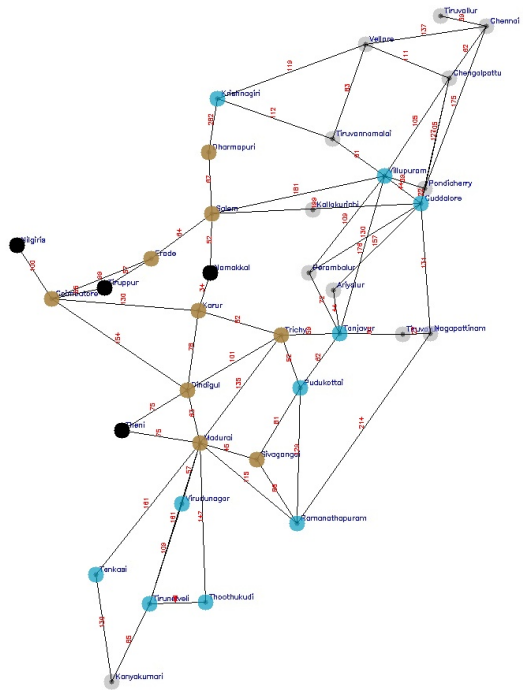
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Pudukottai' - 264,'Tanjavur' - 271,'Ramanathapuram' - 332,'Sivagangai' - 262,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Virudunagar' - 274,'Krishnagiri' - 512,]



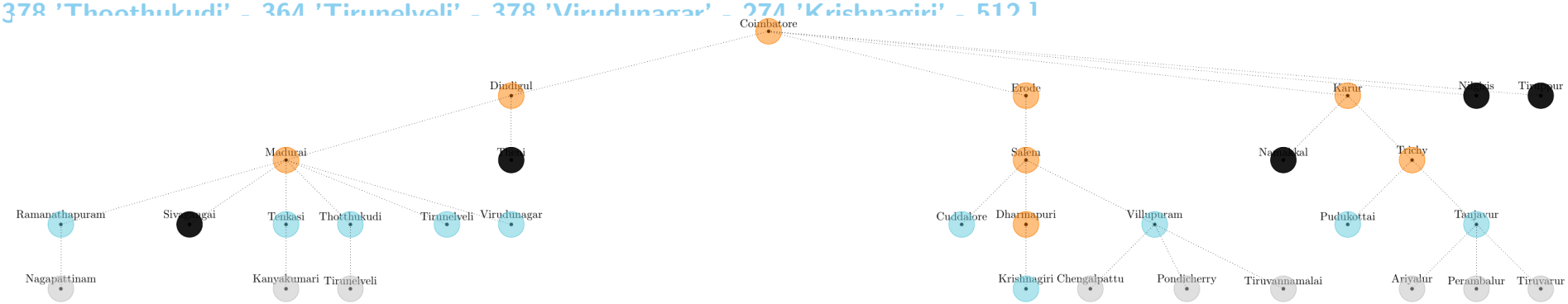


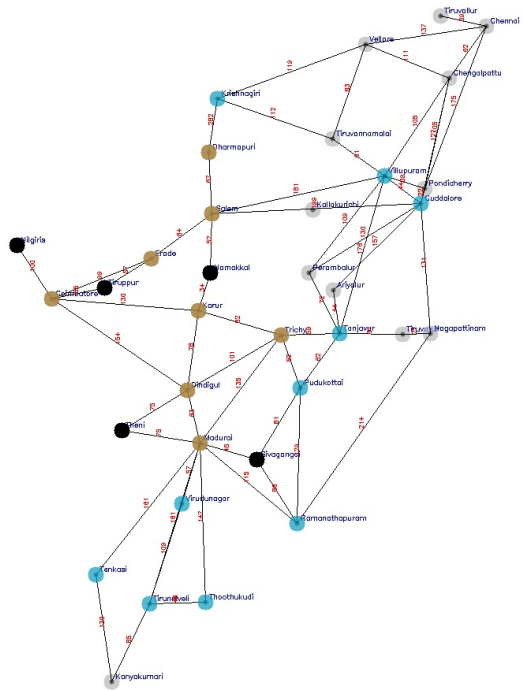
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Pudukottai' - 264,'Tanjavur' - 271,'Ramanathapuram' - 332,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Virudunagar' - 274,'Krishnagiri' - 512,]



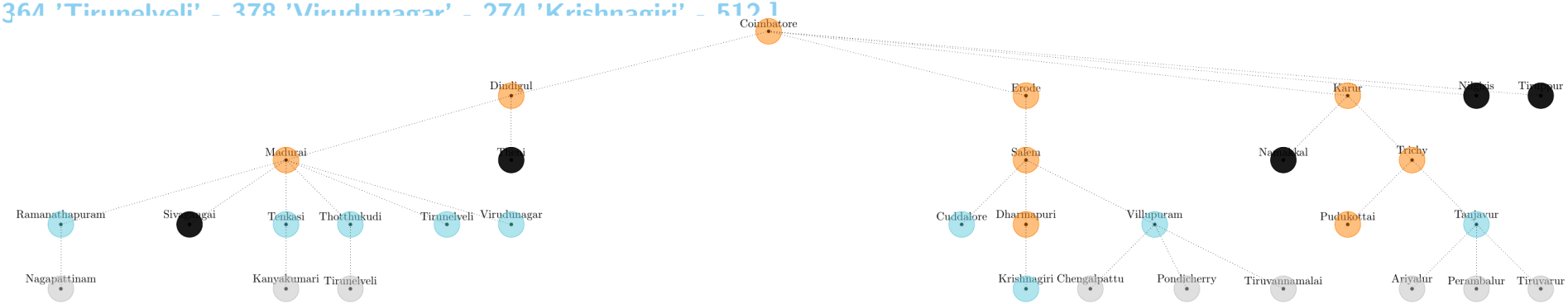


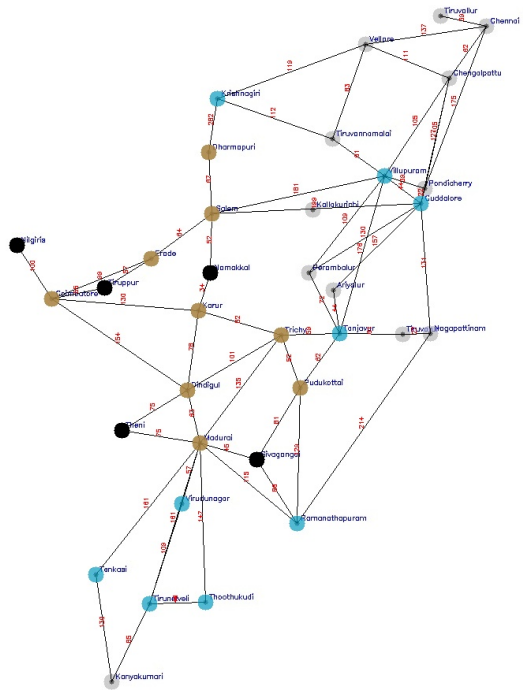
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Pudukottai' - 264,'Tanjavur' - 271,'Ramanathapuram' - 332,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Virudunagar' - 274,'Krishnagiri' - 512,]



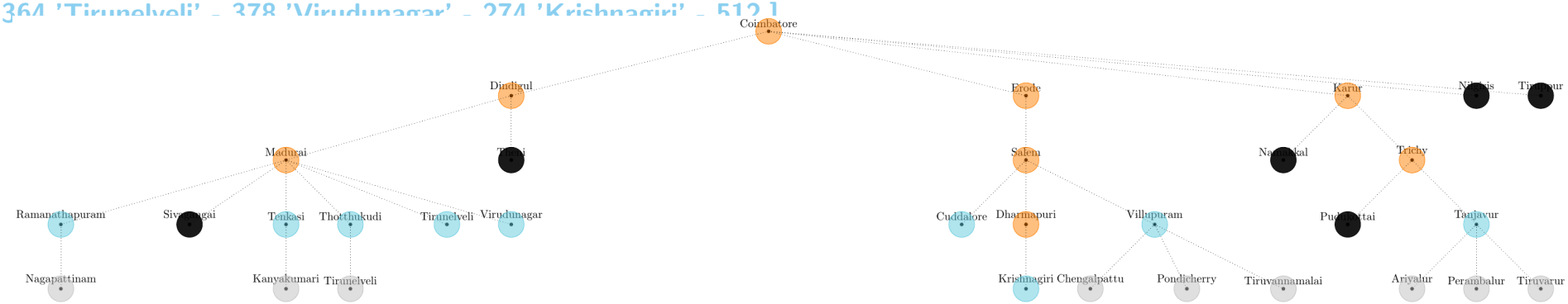


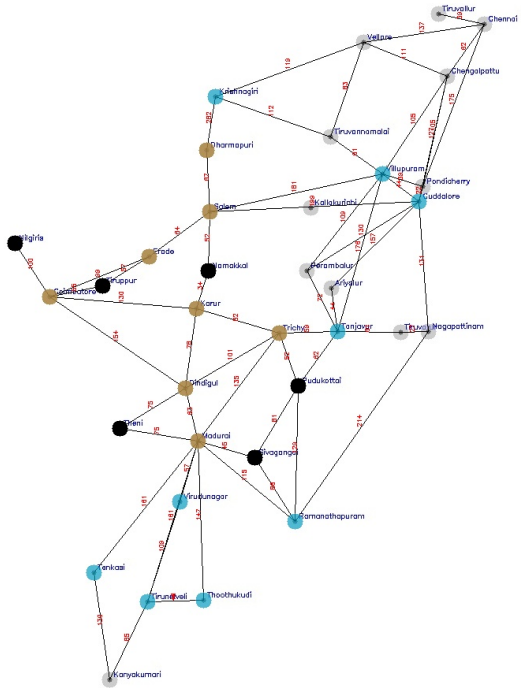
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Tanjavur' - 271,'Ramanathapuram' - 332,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Virudunagar' - 274,'Krishnagiri' - 512,]





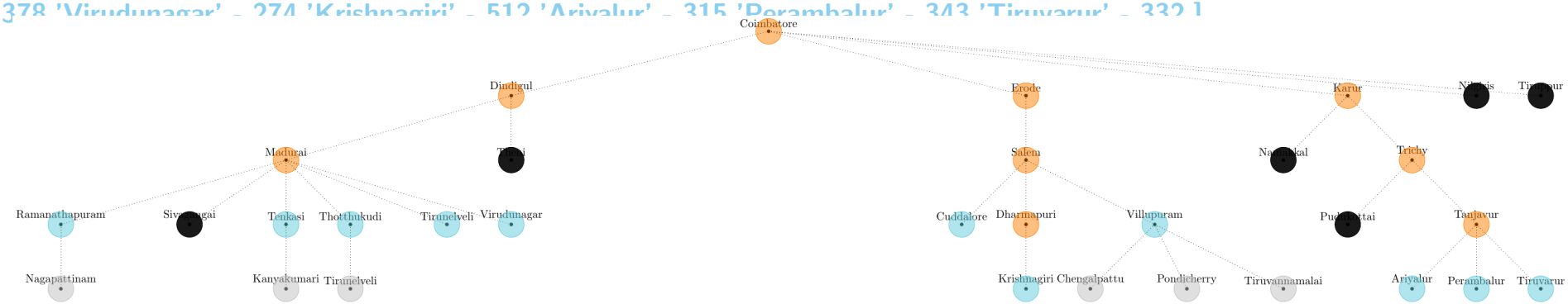
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Tanjavur' - 271,'Ramanathapuram' - 332,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Virudunagar' - 274,'Krishnagiri' - 512,]

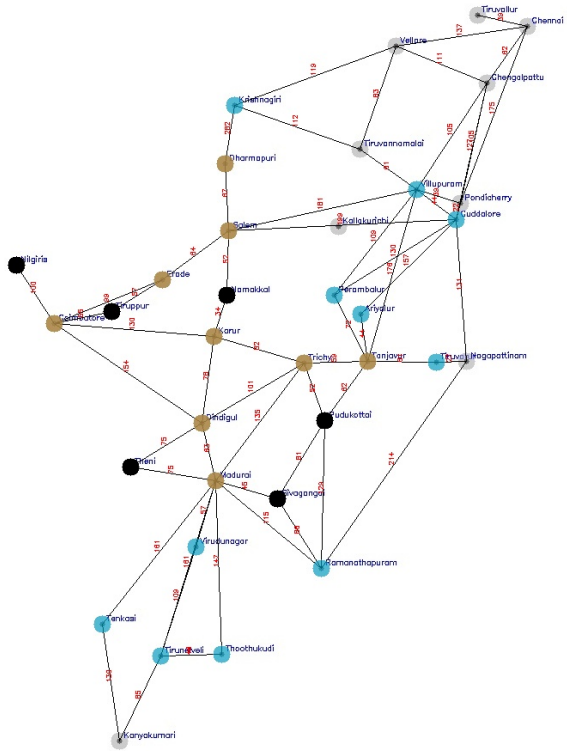




Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Ramanathapuram' - 332,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' -

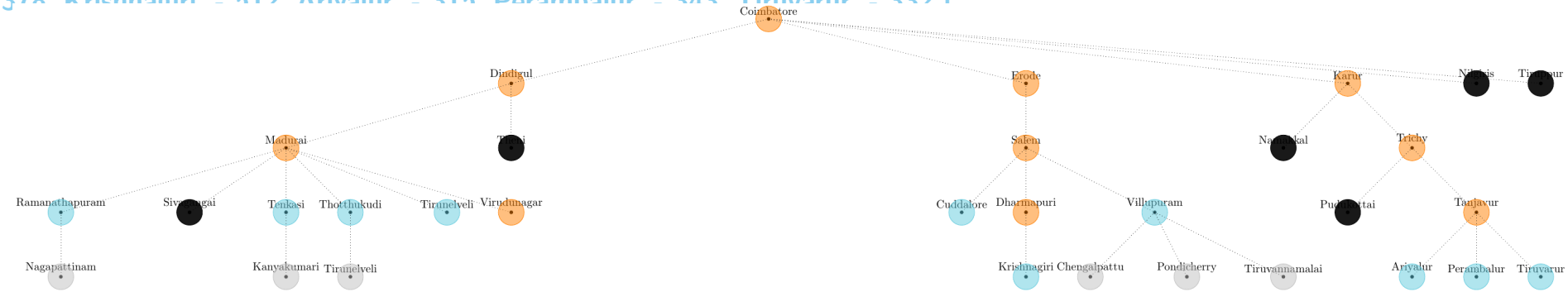
378 'Virudunagar' - 274 'Krishnagiri' - 512 'Ariyalur' - 315 'Perambalur' - 343 'Tiruvarur' - 332]

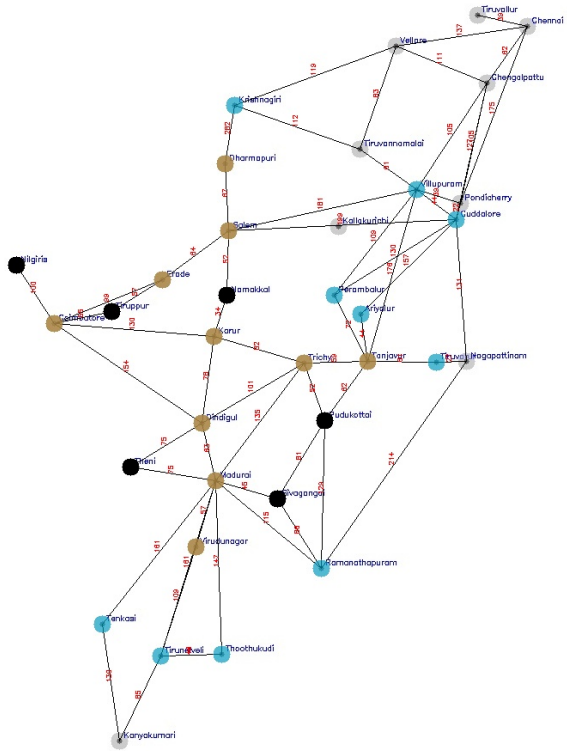




Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Ramanathapuram' - 332,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' -

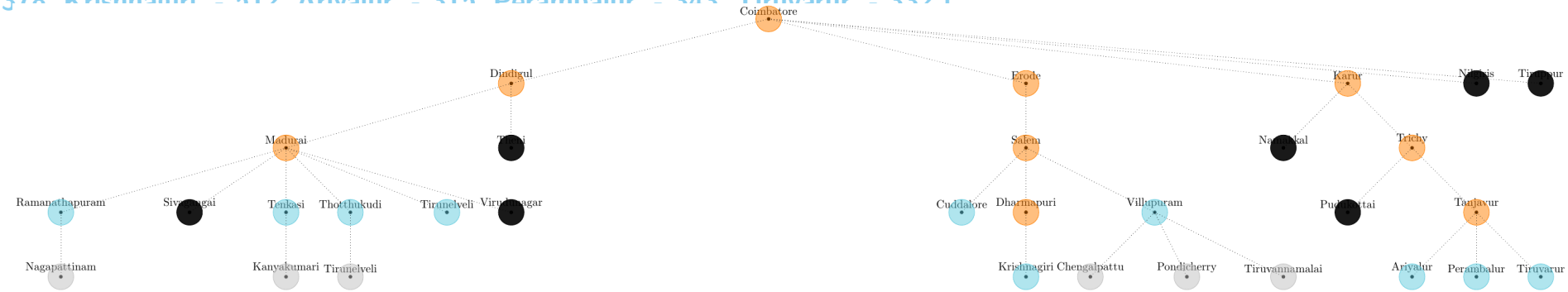
378 'Krishnagiri' - 512 'Arivalur' - 315 'Perambalur' - 343 'Tiruvarur' - 332]

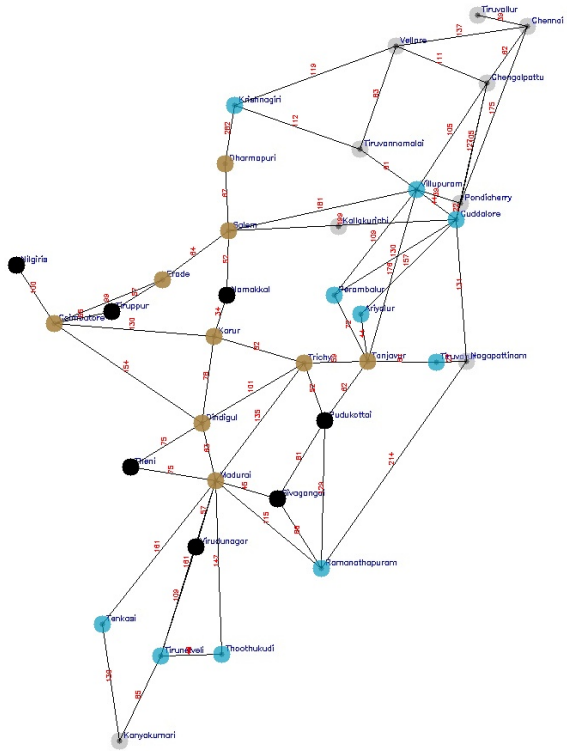




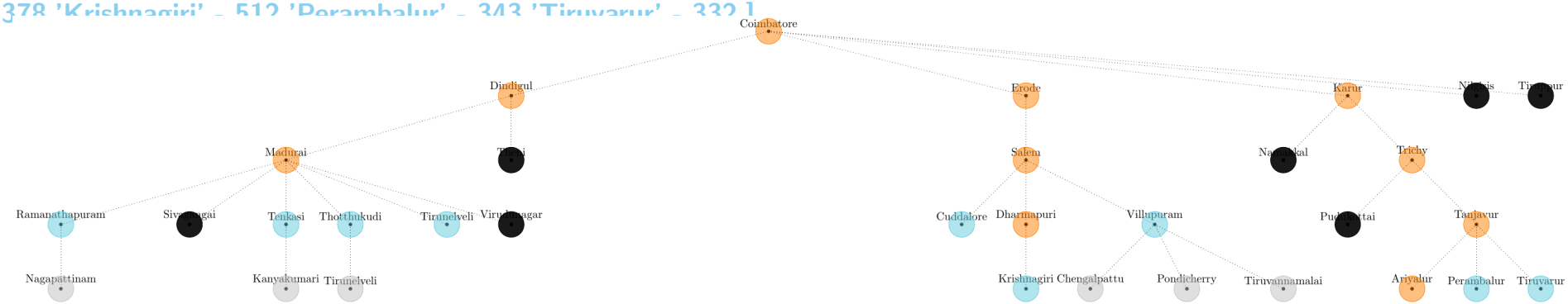
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Ramanathapuram' - 332,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' -

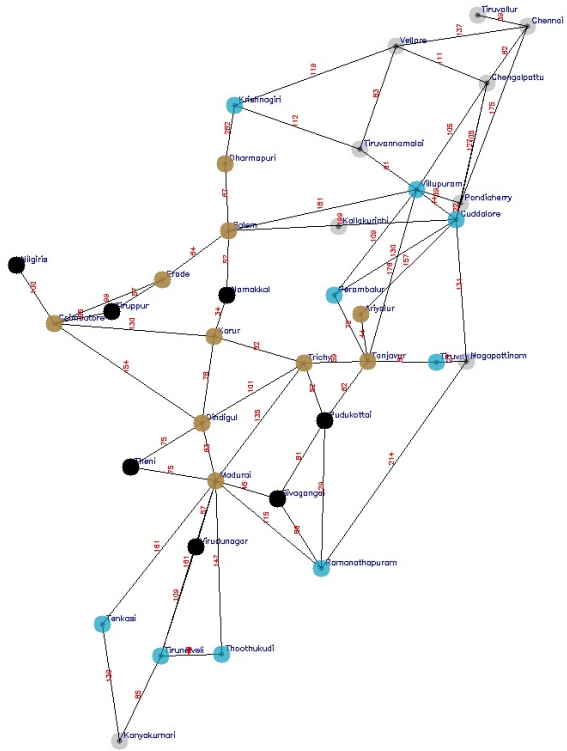
378 'Krishnagiri' - 512 'Arivalur' - 315 'Perambalur' - 343 'Tiruvarur' - 332]



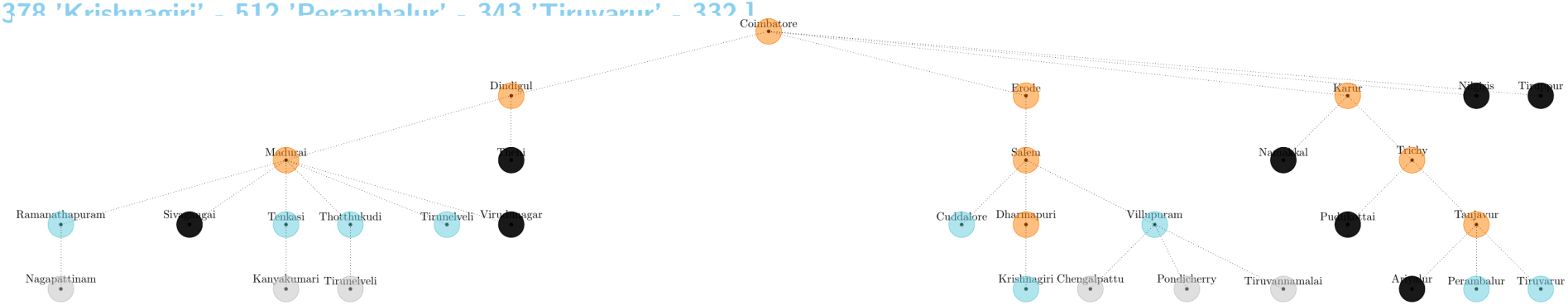


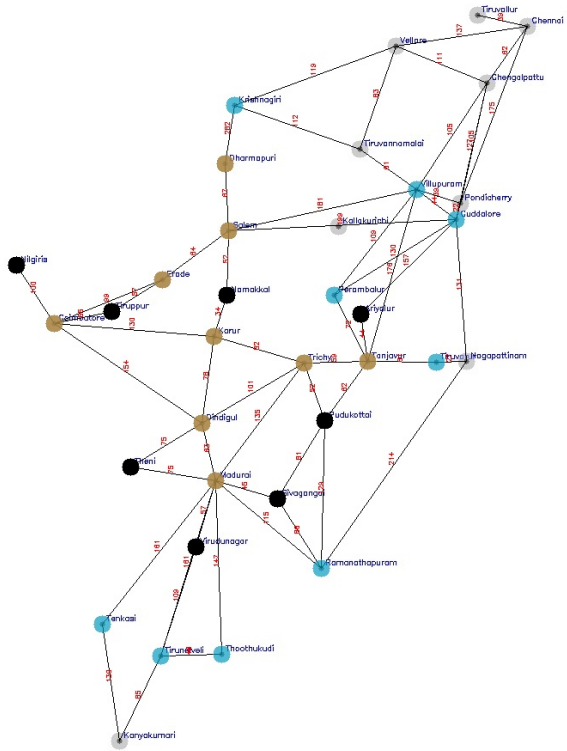
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Ramanathapuram' - 332,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Krishnagiri' - 512,'Perambalur' - 343,'Tiruvarur' - 332,]



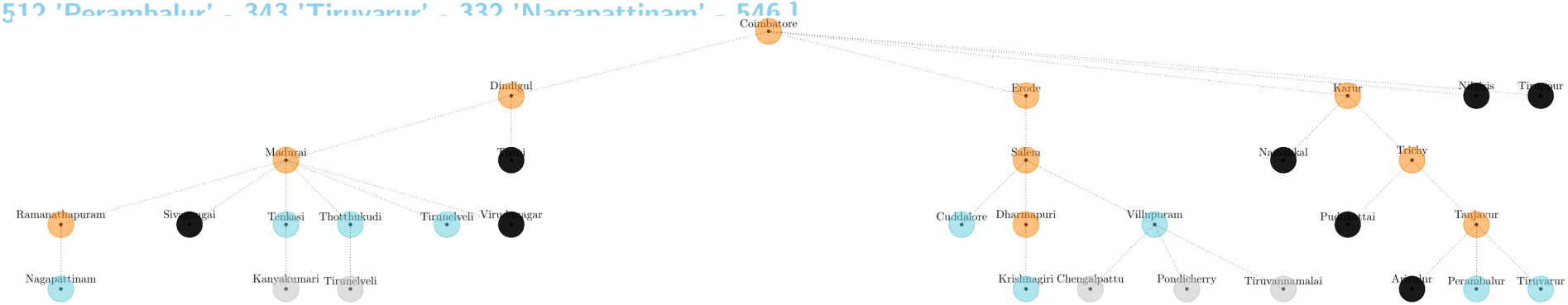


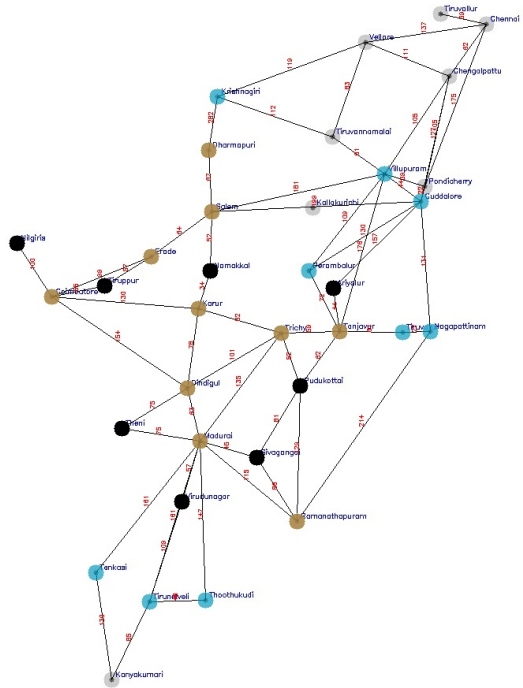
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Ramanathapuram' - 332,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Krishnagiri' - 512,'Perambalur' - 343,'Tiruvarur' - 332,]



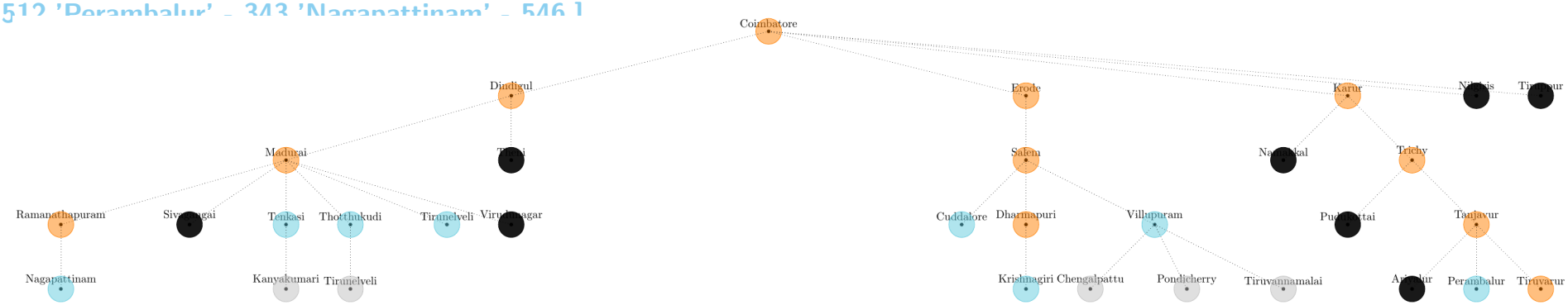


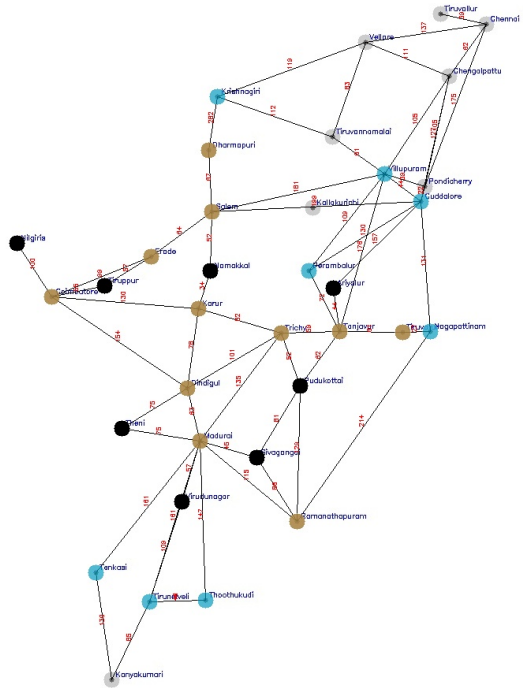
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Krishnagiri' - 512,'Perambalur' - 343,'Tiruvarur' - 332,'Nagapattinam' - 546,]



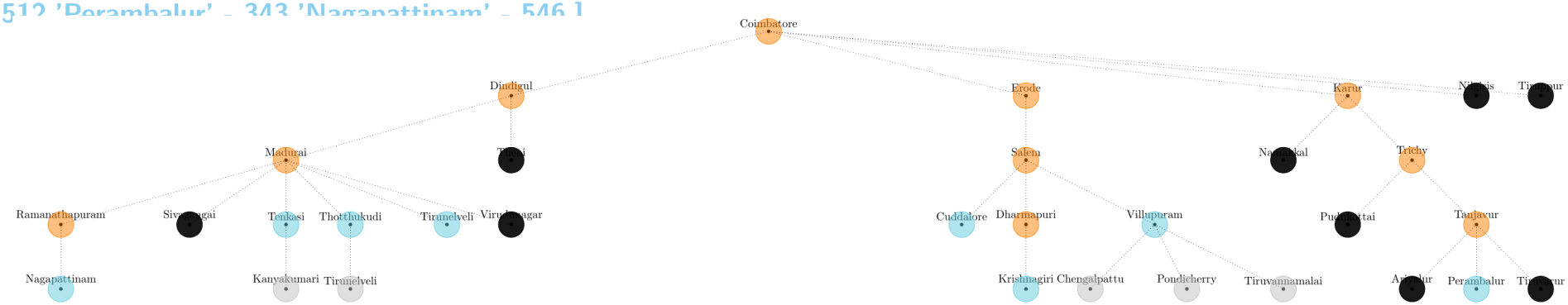


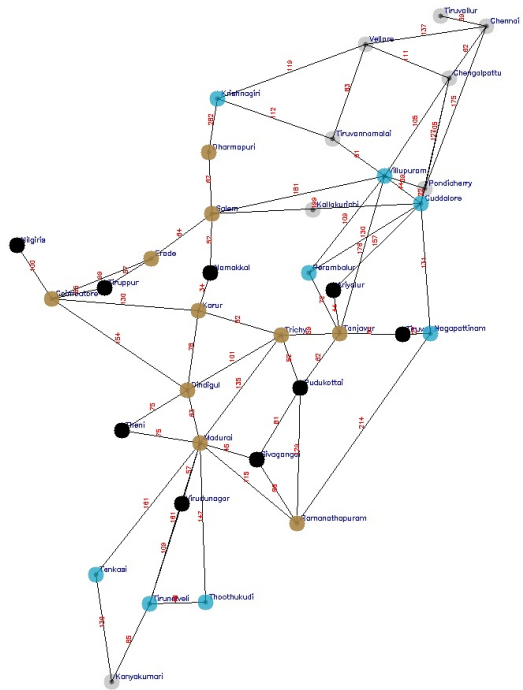
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Krishnagiri' - 512,'Perambalur' - 343,'Nagapattinam' - 546]



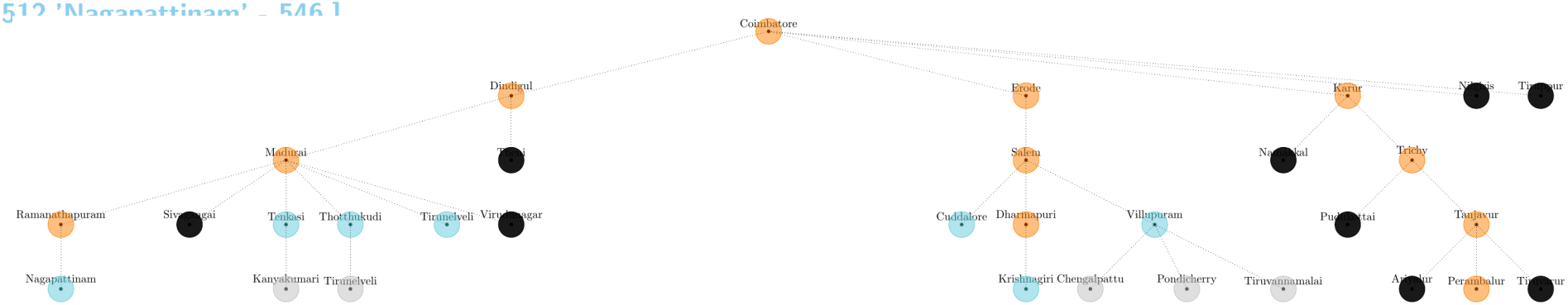


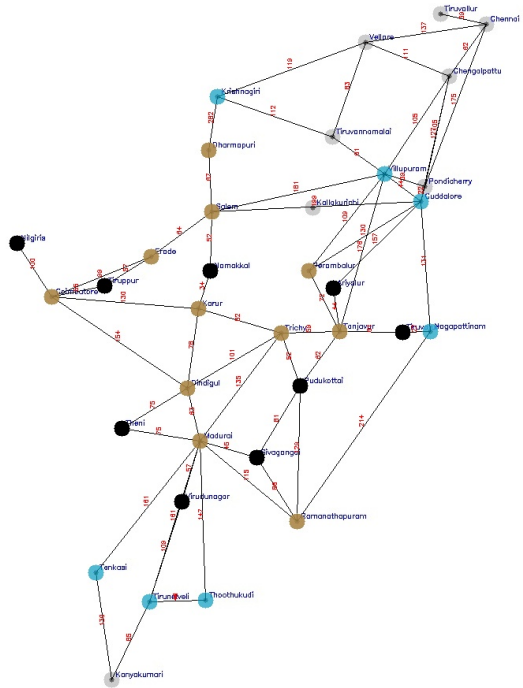
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Krishnagiri' - 512,'Perambalur' - 343,'Nagapattinam' - 546]



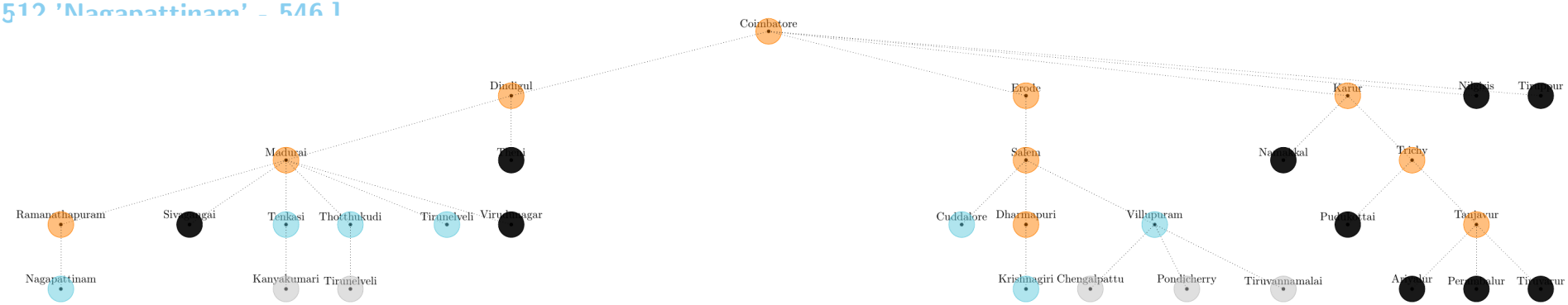


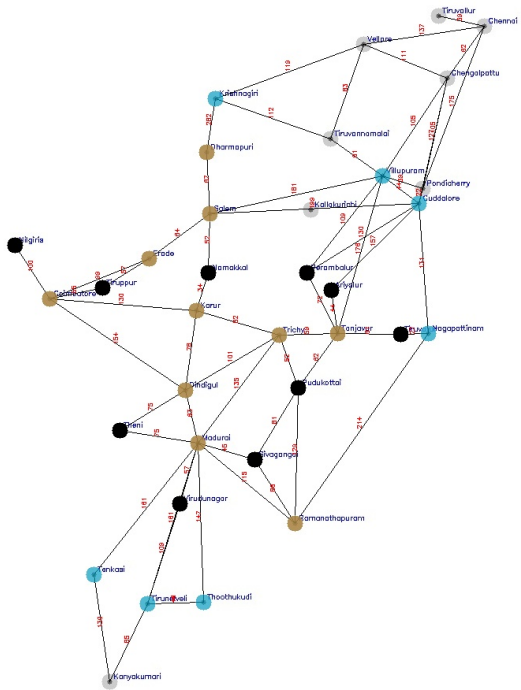
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Krishnagiri' - 512,'Nagapattinam' - 546]



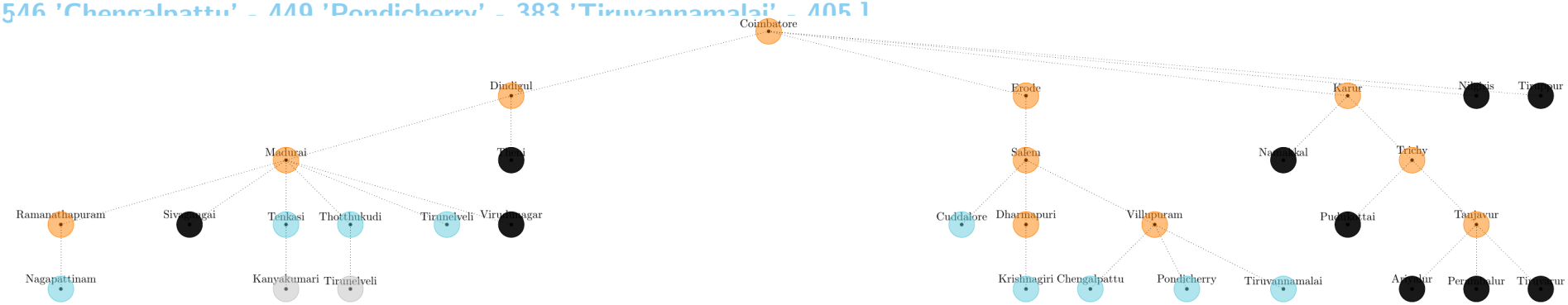


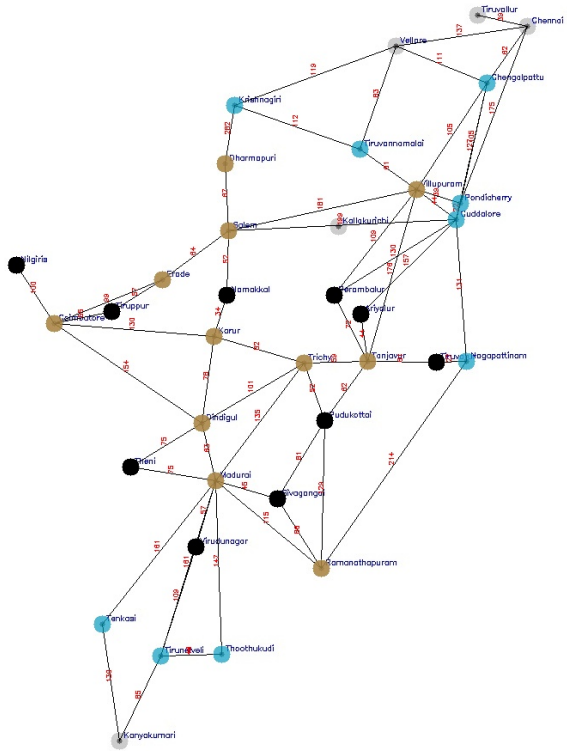
Fontier: ['Cuddalore' - 362,'Villupuram' - 344,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Krishnagiri' - 512,'Nagapattinam' - 546]



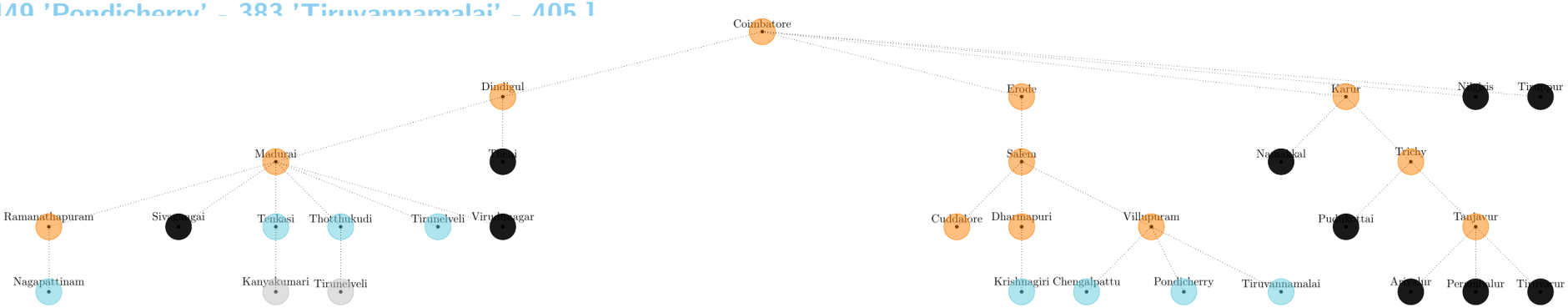


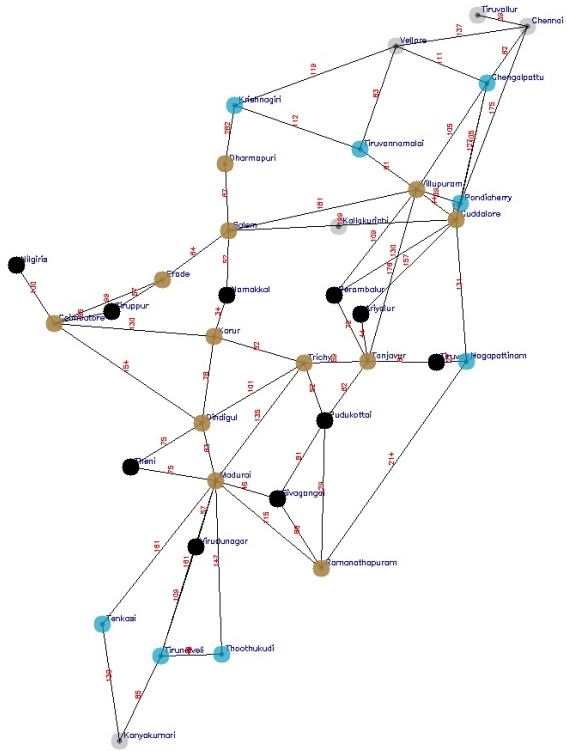
Fontier: ['Cuddalore' - 362,'Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Krishnagiri' - 512,'Nagapattinam' - 546,'Chengalpattu' - 449,'Pondicherry' - 383,'Tiruvannamalai' - 405,]



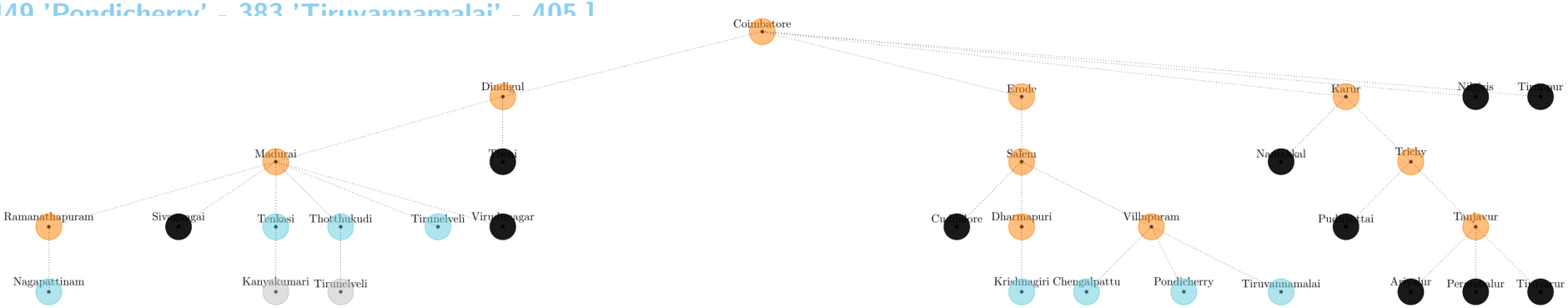


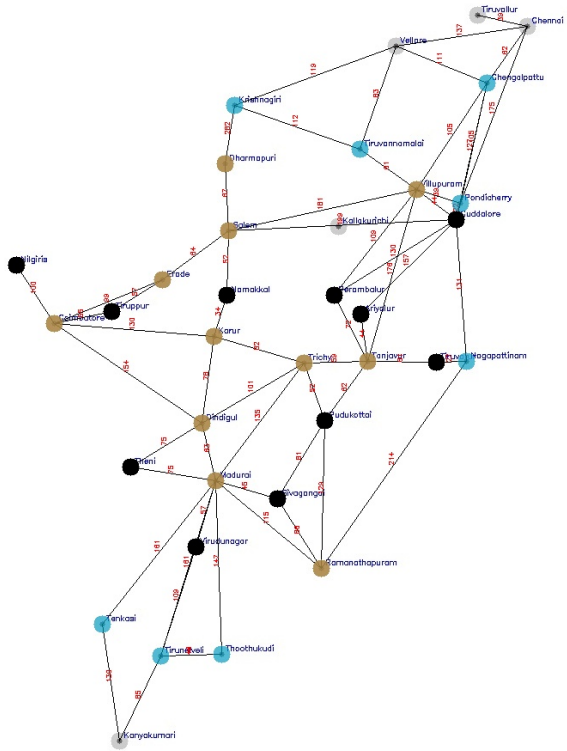
Fontier: ['Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Krishnagiri' - 512,'Nagapattinam' - 546,'Chengalpattu' - 449,'Pondicherry' - 383,'Tiruvannamalai' - 405,]



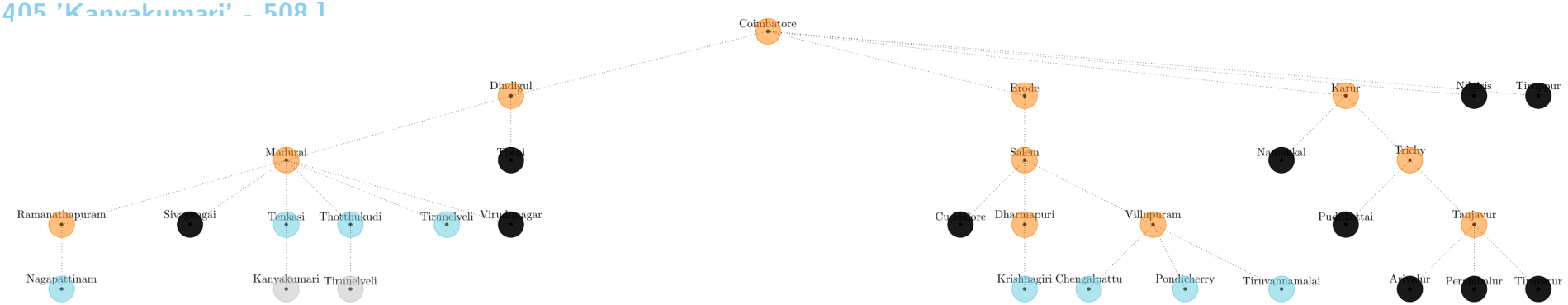


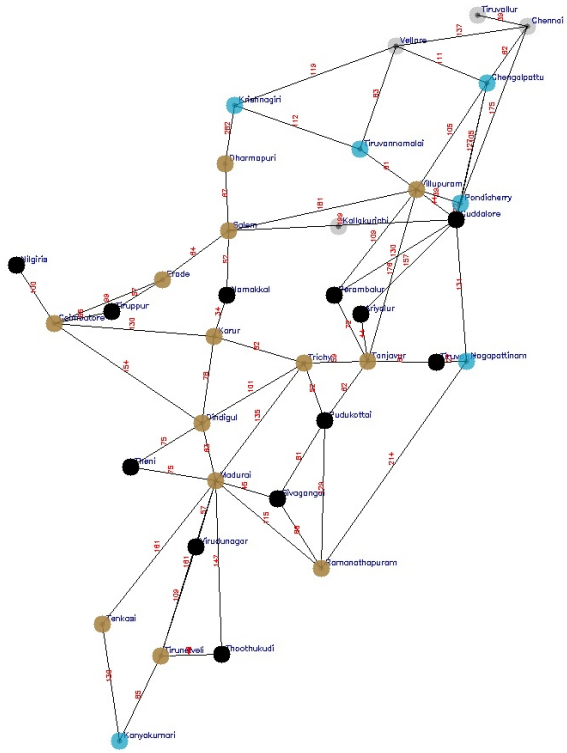
Fontier: ['Tenkasi' - 378,'Thoothukudi' - 364,'Tirunelveli' - 378,'Krishnagiri' - 512,'Nagapattinam' - 546,'Chengalpattu' - 449,'Pondicherry' - 383,'Tiruvannamalai' - 405,]



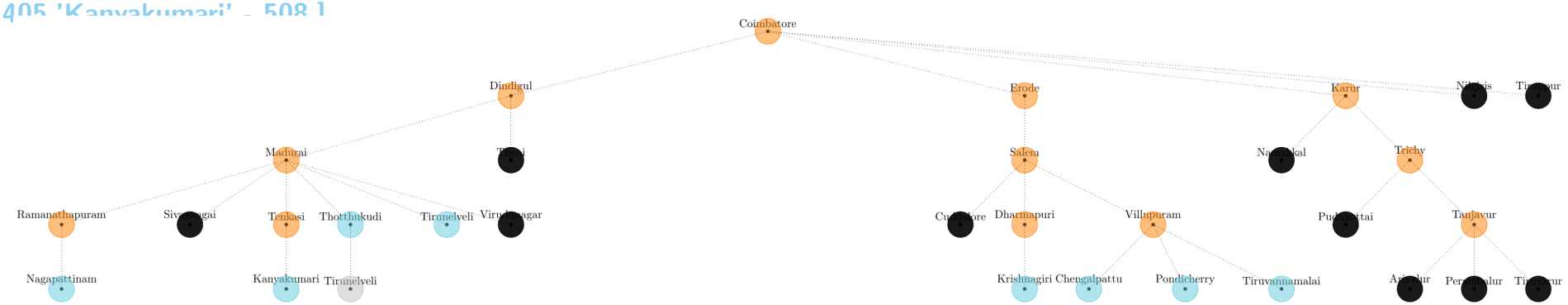


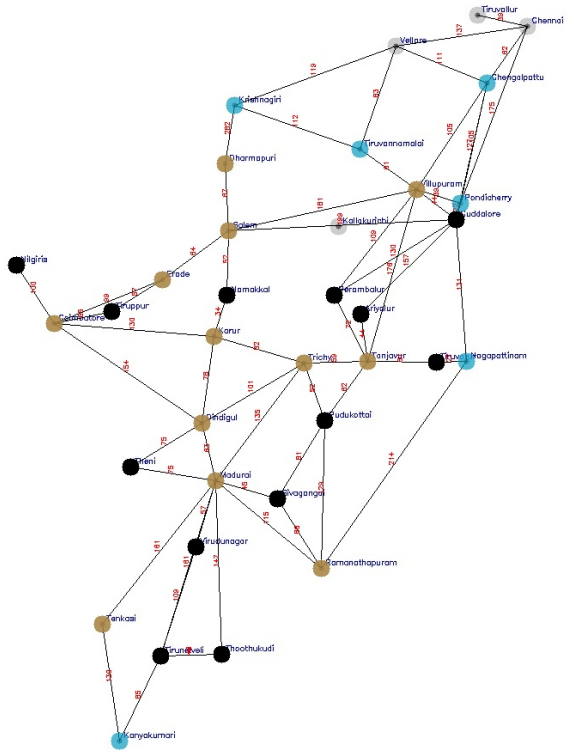
Fontier: ['Krishnagiri' - 512,'Nagapattinam' - 546,'Chengalpattu' - 449,'Pondicherry' - 383,'Tiruvannamalai' - 405,'Kanyakumari' - 508,]



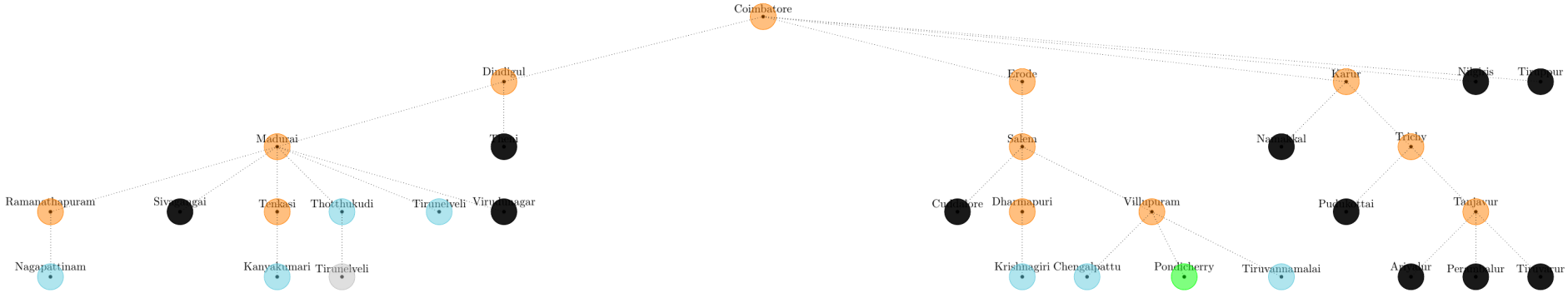


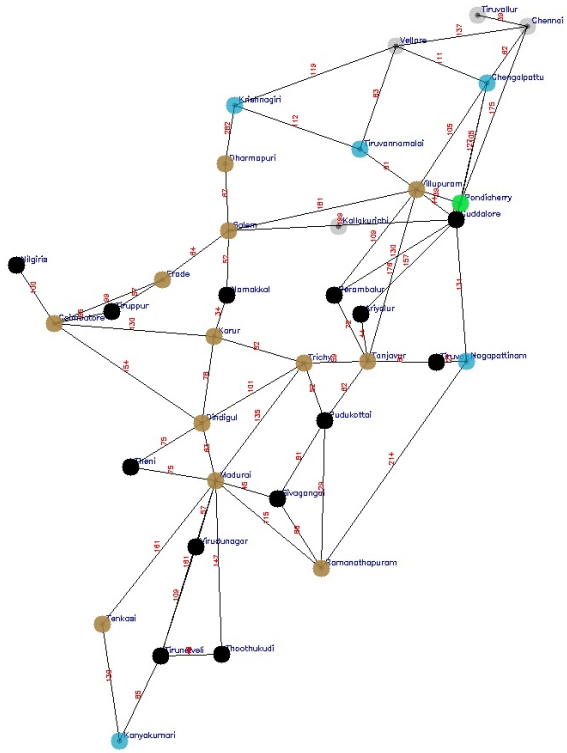
Fontier: ['Krishnagiri' - 512,'Nagapattinam' - 546,'Chengalpattu' - 449,'Pondicherry' - 383,'Tiruvannamalai' - 405,'Kanyakumari' - 508,]



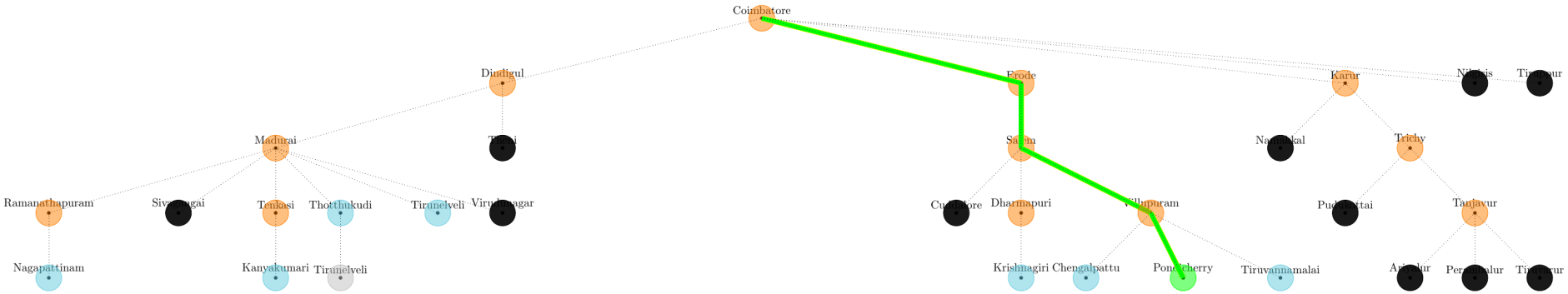


Fontier: ['Krishnagiri' - 512,'Nagapattinam' - 546,'Chengalpattu' - 449,'Tiruvannamalai' - 405,'Kanyakumari' - 508,]





The path UCS took is from Coimbatore to Pondicherry is Coimbatore $->$ Erode $->$ Salem $->$ Villupuram $->$ Pondicherry.

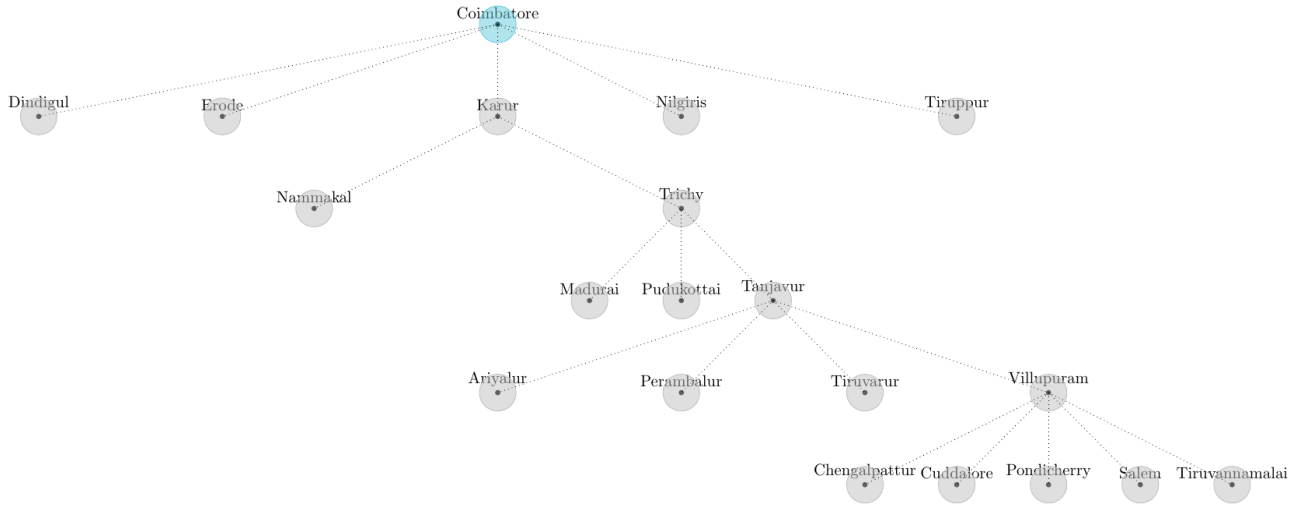


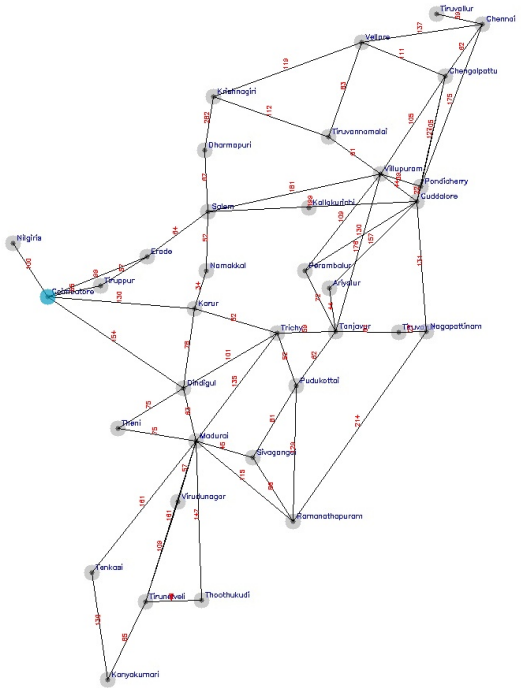
Greedy Search

- In Greedy Search , we expand the node which appears to be closest to goal.
- It uses domain infromation of straight-line distance from b to goal f (n) = h(n).
- The data structure used is a **HEAP**, we use the estimated cost as function.

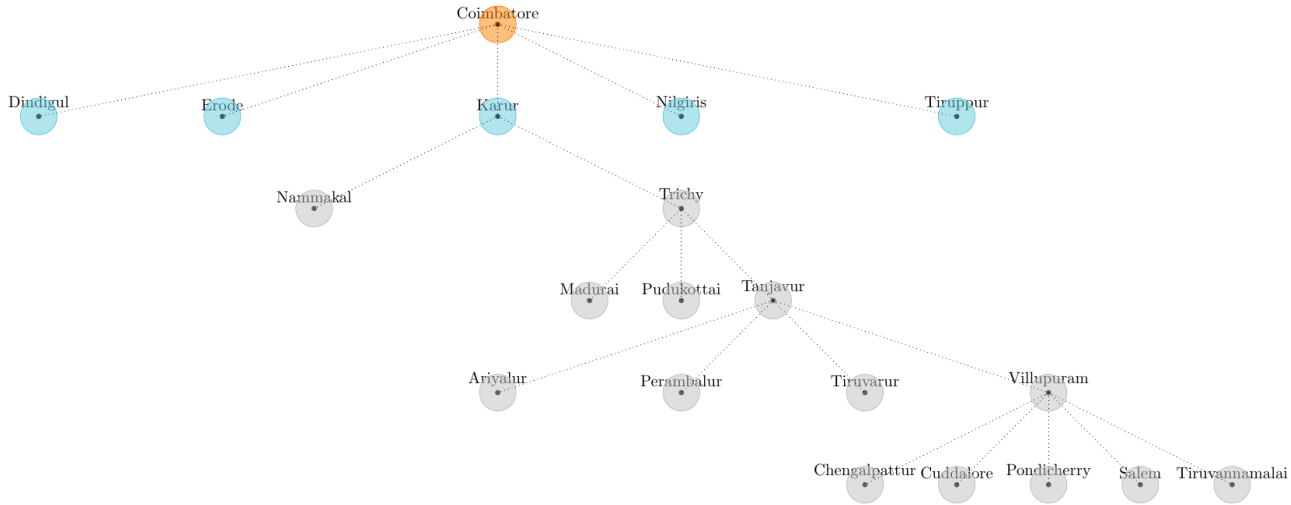
```
function A-STAR-SEARCH(initialState, goalTest)
  returns SUCCESS or FAILURE: /* Cost f(n) =  h(n) */

  frontier = Heap.new(initialState)
  explored = Set.new()

  while not frontier.isEmpty()
    state = frontier.deleteMin()
    explored.add(state)
    if goalTest(state):
      return SUCCESS(state)

    for neighbor in state.neighbors():
      if neighbor not in frontier U explored:
        frontier.insert(neighbor)
      else if neighbor in frontier:
        frontier.decreaseKey(neighbor)
  return FAILURE
```

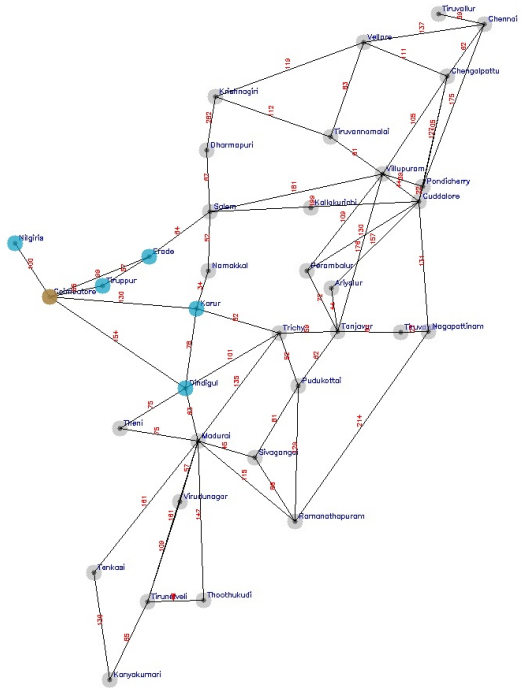
Fontier: ['Coimbatore - 492']



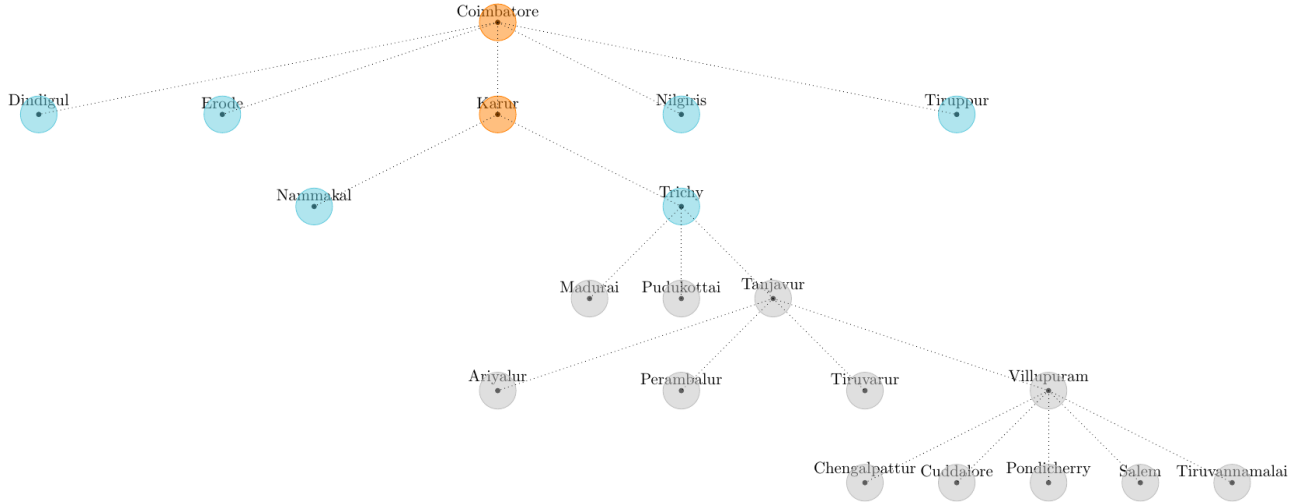


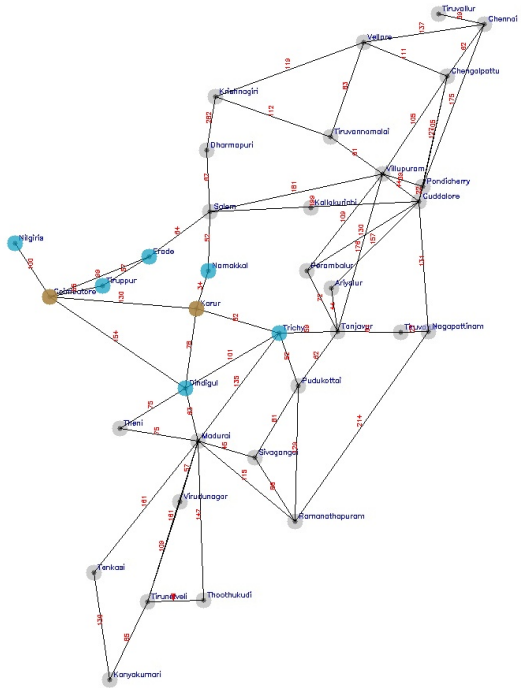
Fontier: ['Dindigul' - 395,'Erode' - 358,'Karur' - 326,'Nilgiris' - 521,'Tiruppur' - 425,]



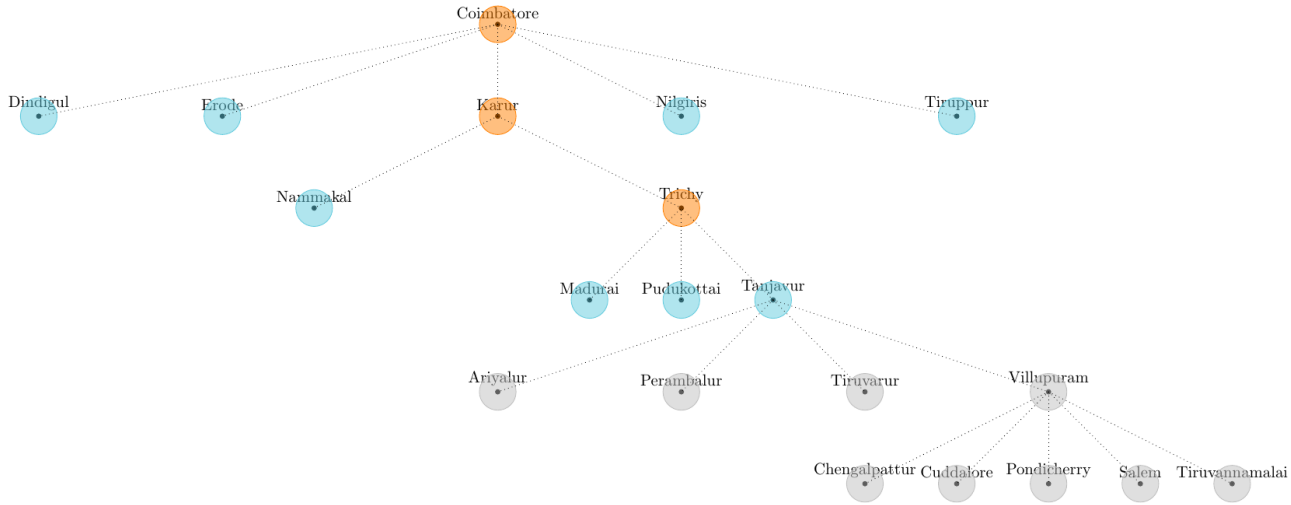


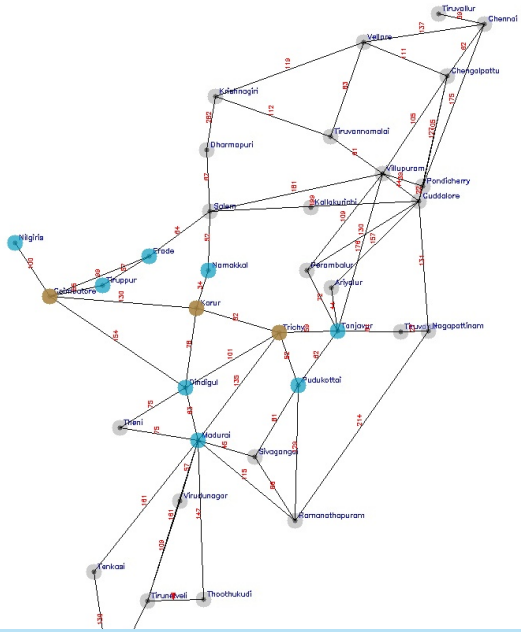
Fontier: ['Dindigul' - 395,'Erode' - 358,'Nilgiris' - 521,'Tiruppur' - 425,'Namakkal' - 292,'Trichy' - 260,]



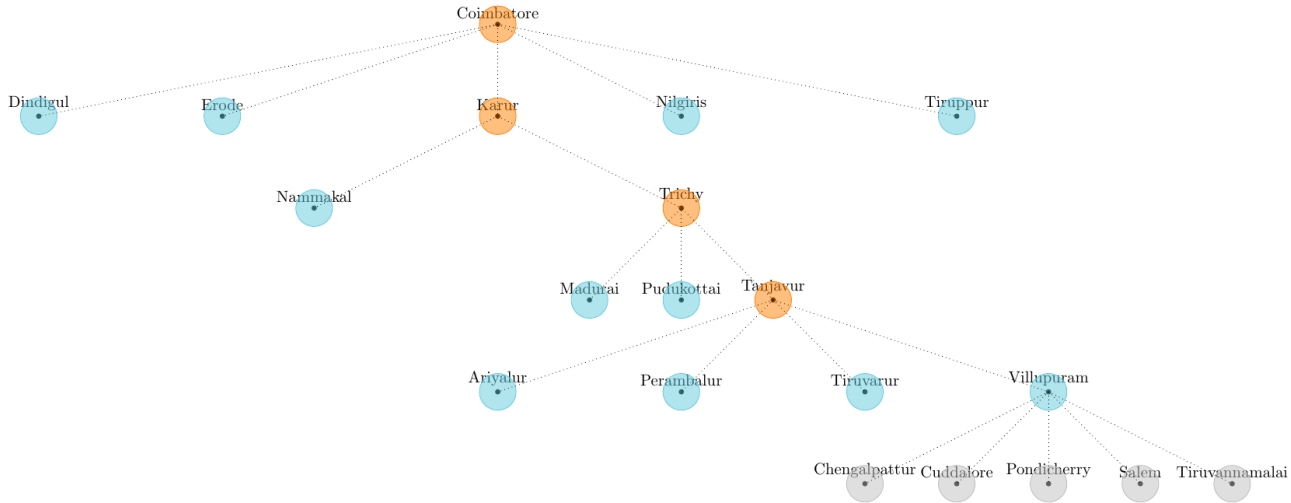


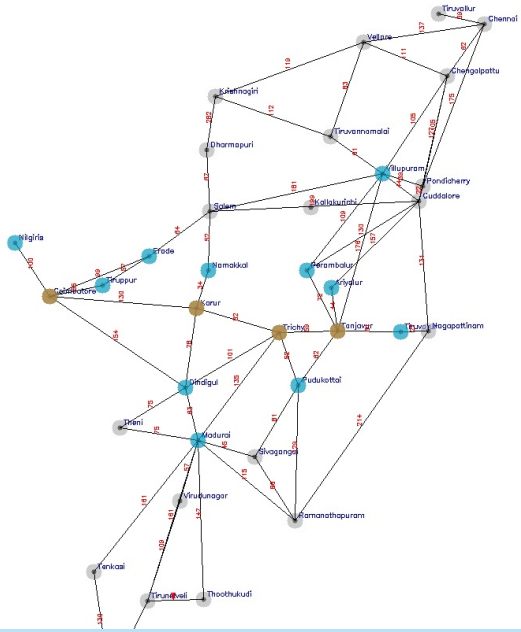
Fontier: ['Dindigul' - 395,'Erode' - 358,'Nilgiris' - 521,'Tiruppur' - 425,'Namakkal' - 292,'Madurai' - 430,'Pudukottai' - 298,'Tanjavur' - 213,]



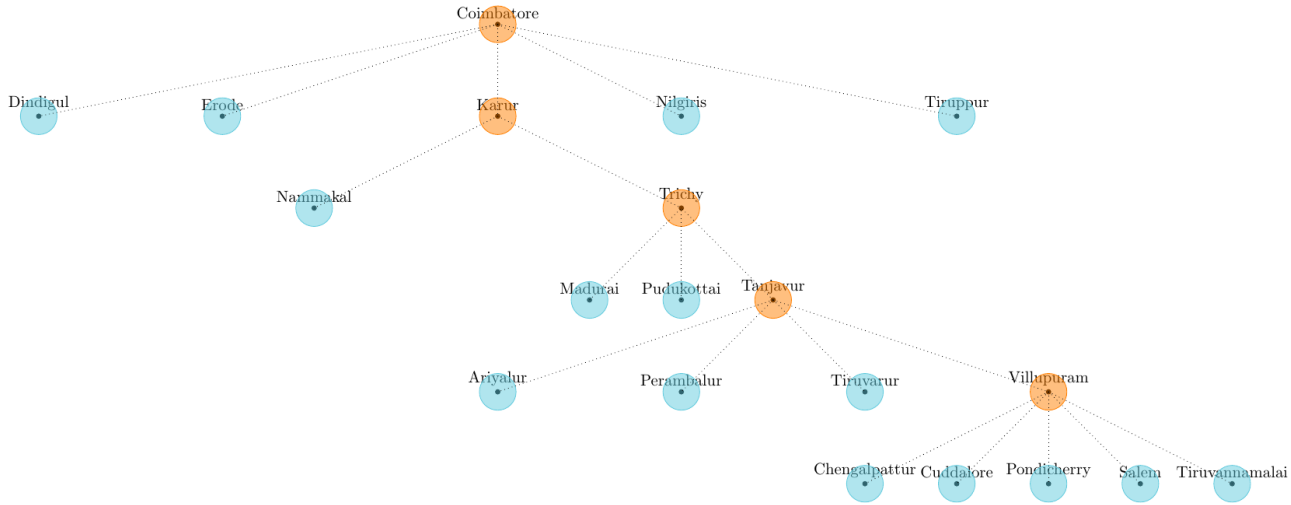


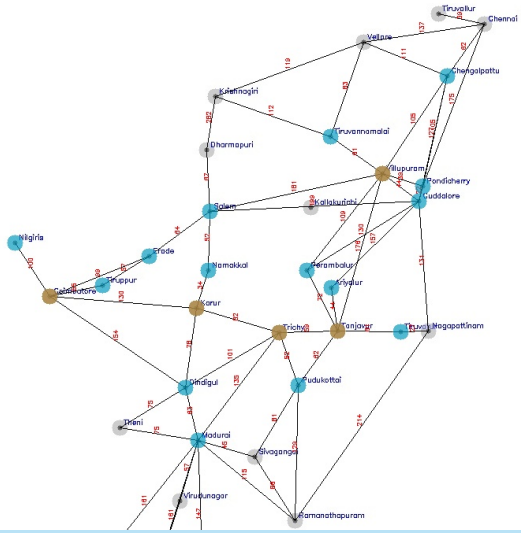
Fontier: ['Dindigul' - 395,'Erode' - 358,'Nilgiris' - 521,'Tiruppur' - 425,'Namakkal' - 292,'Madurai' - 430,'Pudukottai' - 298,'Ariyalur' - 173,'Perambalur' - 181,'Tiruvarur' - 187,'Villupuram' - 53,]



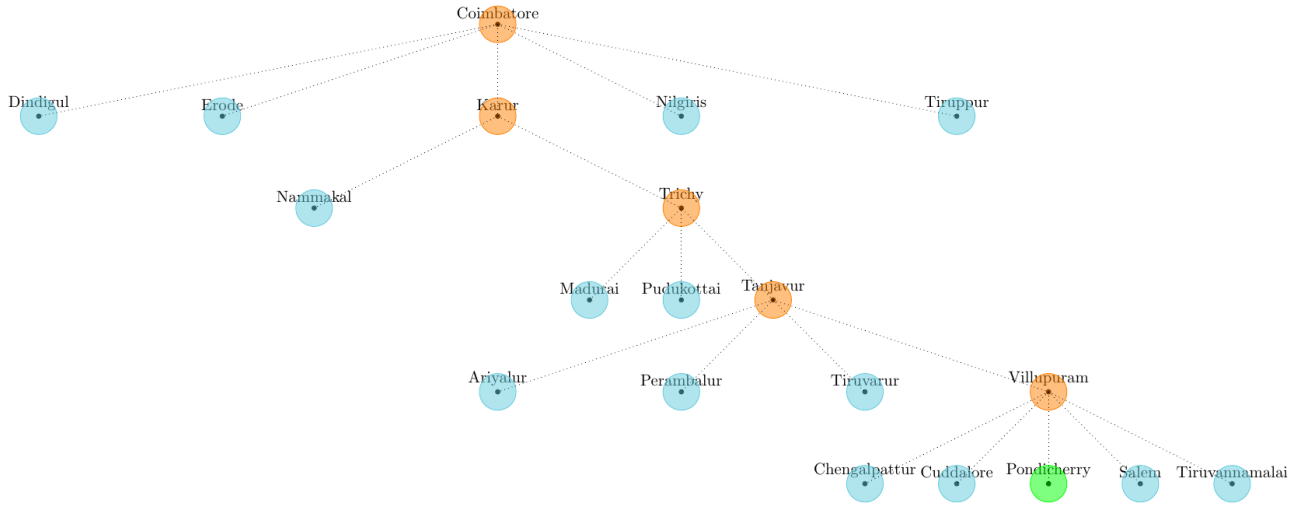


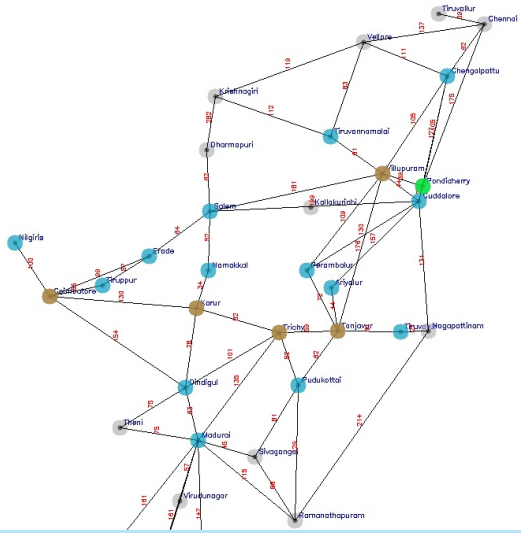
Fontier: ['Dindigul' - 395,'Erode' - 358,'Nilgiris' - 521,'Tiruppur' - 425,'Namakkal' - 292,'Madurai' - 430,'Pudukottai' - 298,'Ariyalur' - 173,'Perambalur' - 181,'Tiruvarur' - 187,'Chengalpattu' - 143,'Cuddalore' - 19,'Pondicherry' - 0,'Salem' - 271,'Tiruvannamalai' - 132,]



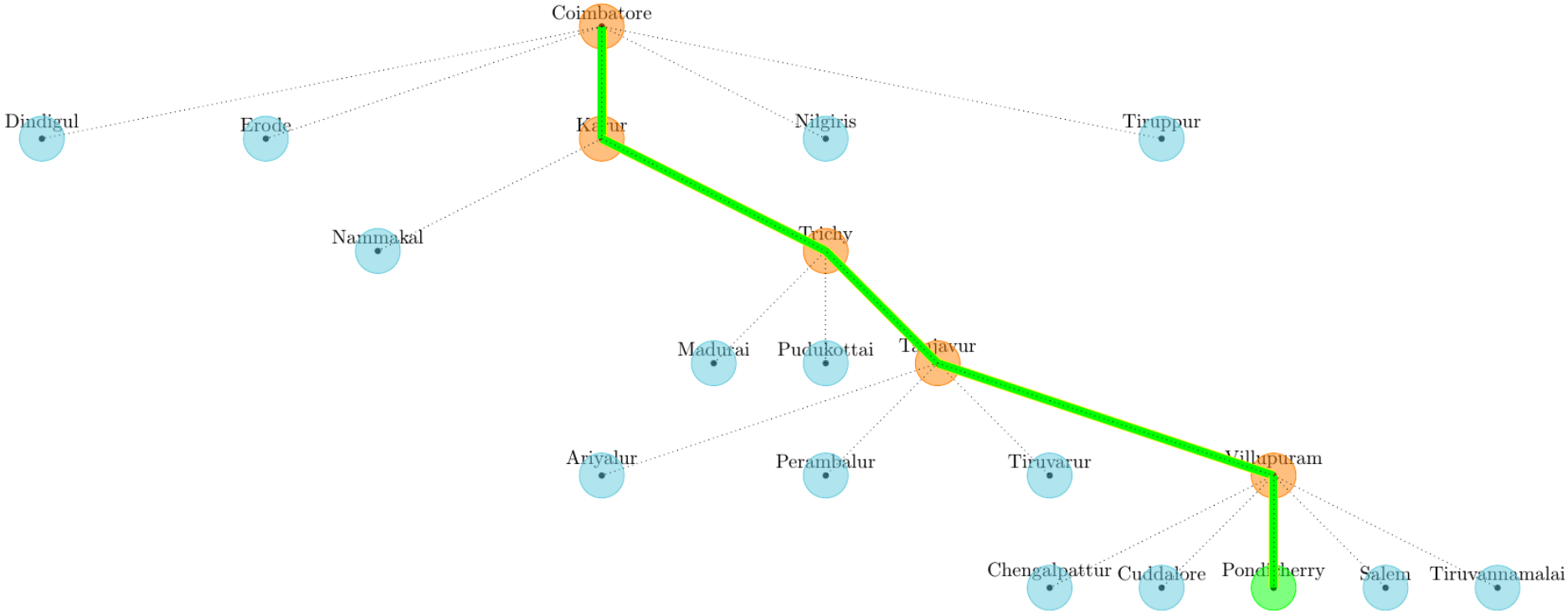


Fontier: ['Dindigul' - 395,'Erode' - 358,'Nilgiris' - 521,'Tiruppur' - 425,'Namakkal' - 292,'Madurai' - 430,'Pudukottai' - 298,'Ariyalur' - 173,'Perambalur' - 181,'Tiruvarur' - 187,'Chengalpattu' - 143,'Cuddalore' - 19,'Salem' - 271,'Tiruvannamalai' - 132,]





The path Greedy Search took is from Coimbatore to Pondicherry is Coimbatore $->$ Karur $->$ Trichy $->$ Tanjavur $->$ Villupuram $->$ Pondicherry.

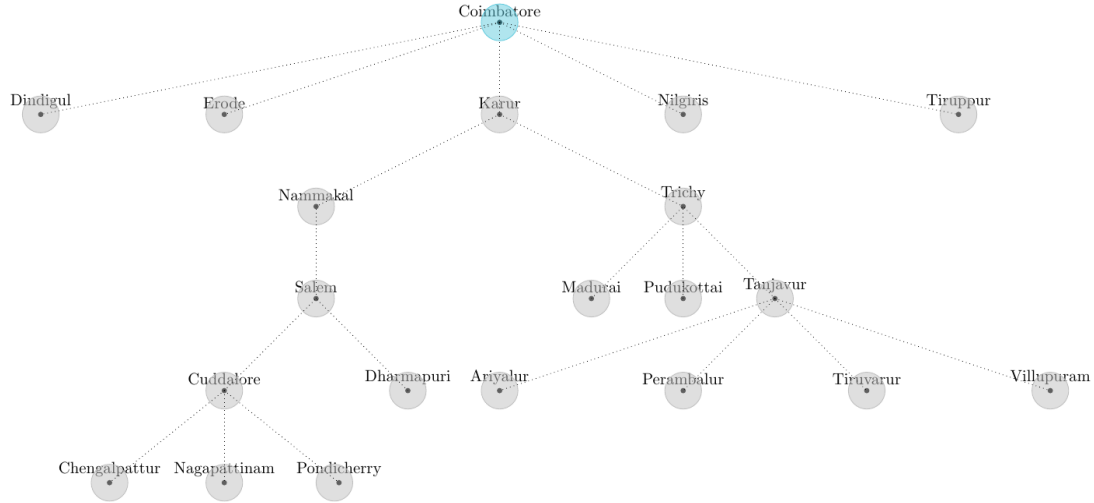


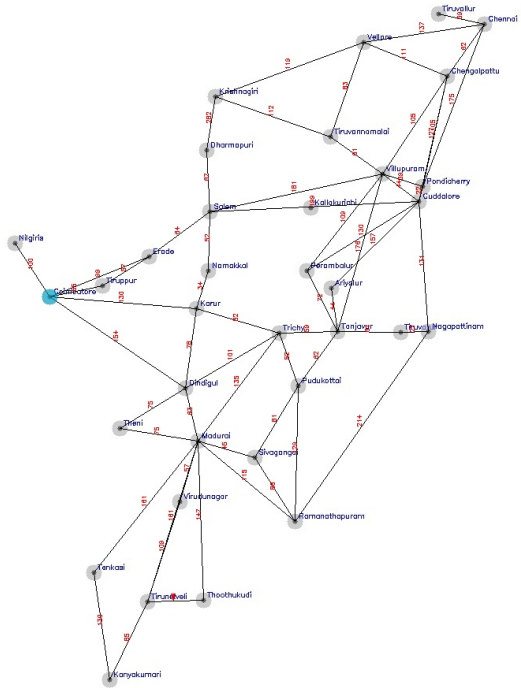
A^* Search

- In A^* Search , we expand the node which Minimize the total estimated solution cost.
- It uses domain infromation of cost to reach node n and cost to get from n to goal f (n) = g(n) + h(n).
- The data structure used is a **HEAP**, we use the estimated cost as function.

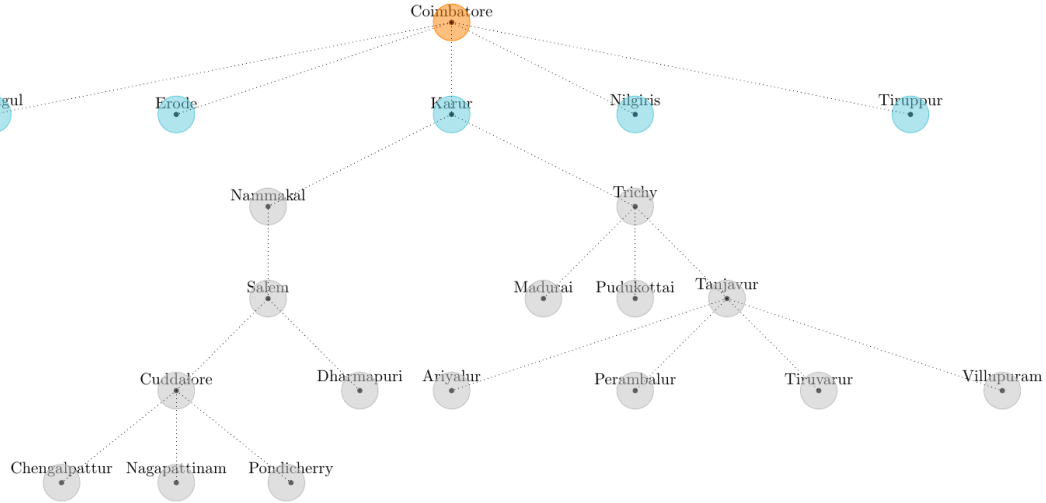
```
function A-STAR-SEARCH(initialState, goalTest)
  returns SUCCESS or FAILURE: /* Cost f(n) = g(n) + h(n) */

  frontier = Heap.new(initialState)
  explored = Set.new()

  while not frontier.isEmpty()
    state = frontier.deleteMin()
    explored.add(state)
    if goalTest(state):
      return SUCCESS(state)

    for neighbor in state.neighbors():
      if neighbor not in frontier U explored:
        frontier.insert(neighbor)
      else if neighbor in frontier:
        frontier.decreaseKey(neighbor)
  return FAILURE
```

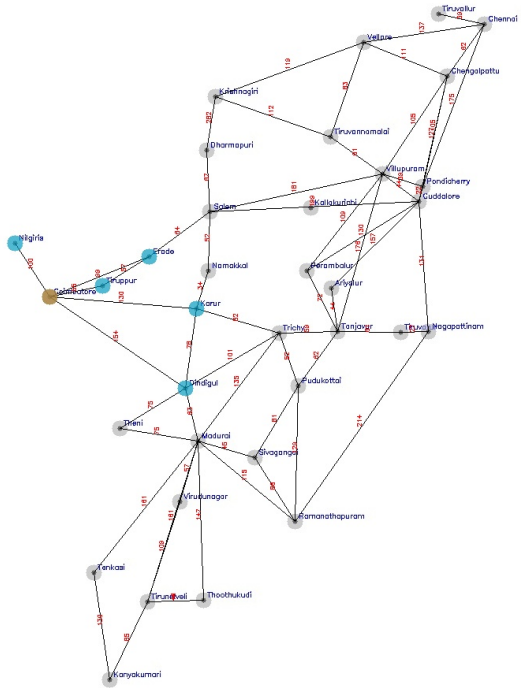
Fontier: ['Coimbatore - 492']



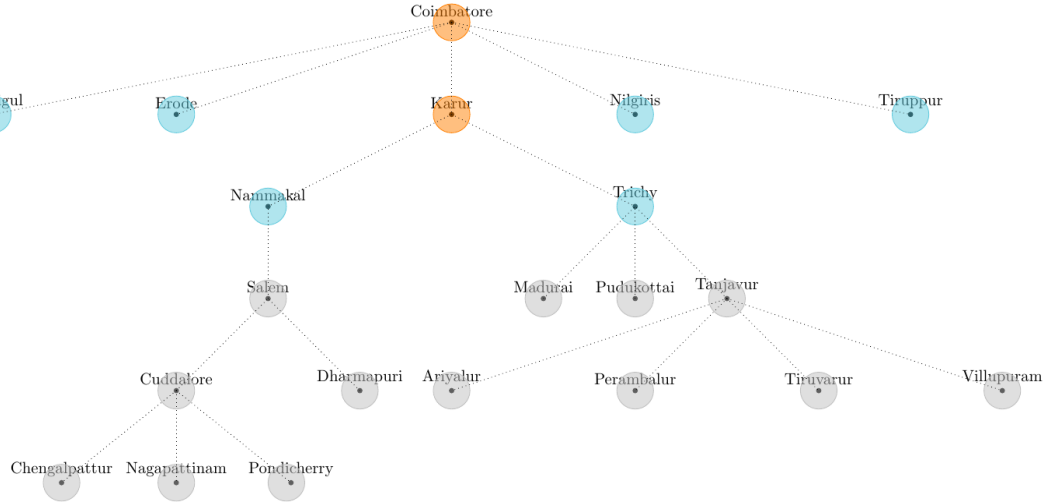


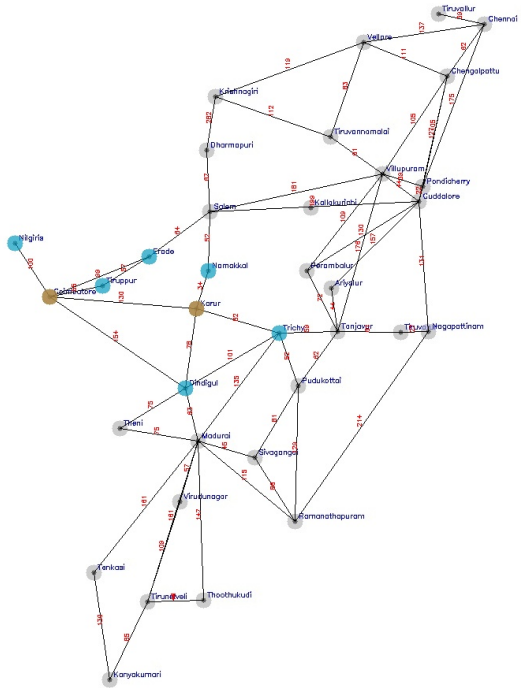
Fontier: ['Dindigul' - 549,'Erode' - 457,'Karur' - 456,'Nilgiris' - 621,'Tiruppur' - 480,]



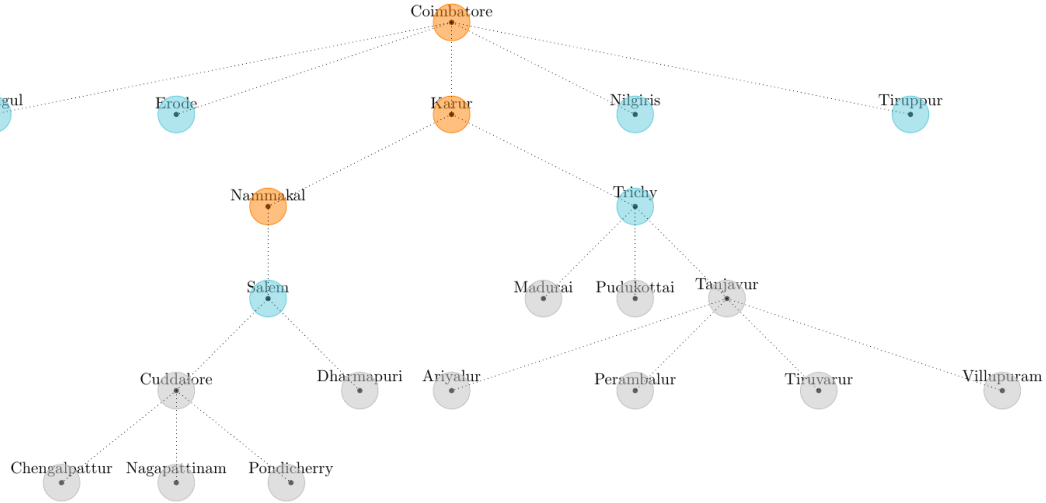


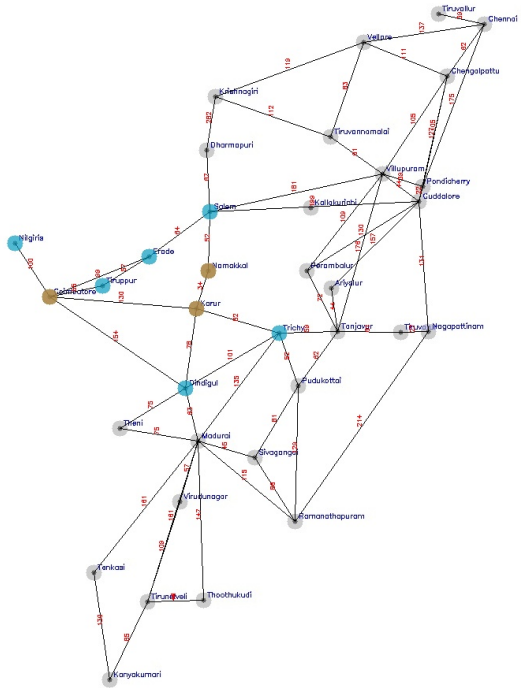
Fontier: ['Dindigul' - 549,'Erode' - 457,'Nilgiris' - 621,'Tiruppur' - 480,'Namakkal' - 456,'Trichy' - 472,]



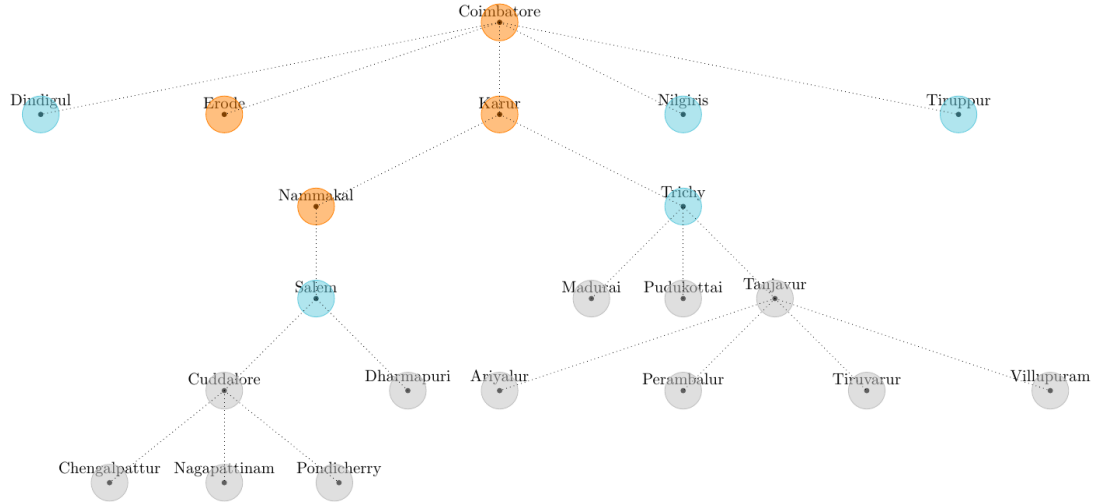


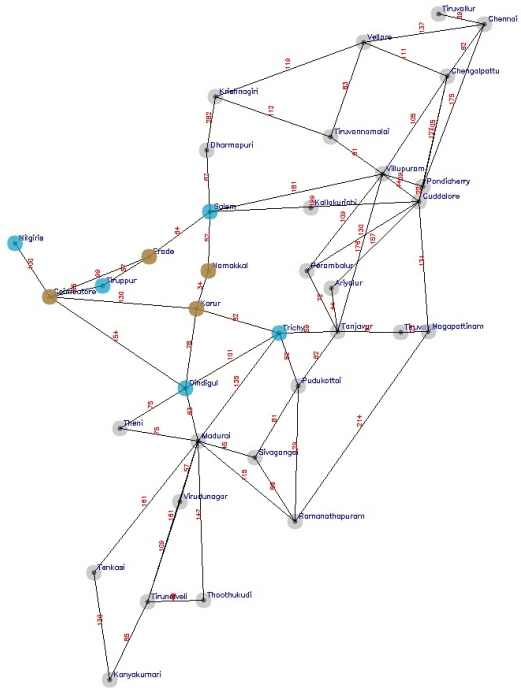
Fontier: ['Dindigul' - 549,'Erode' - 457,'Nilgiris' - 621,'Tiruppur' - 480,'Trichy' - 472,'Salem' - 487,]



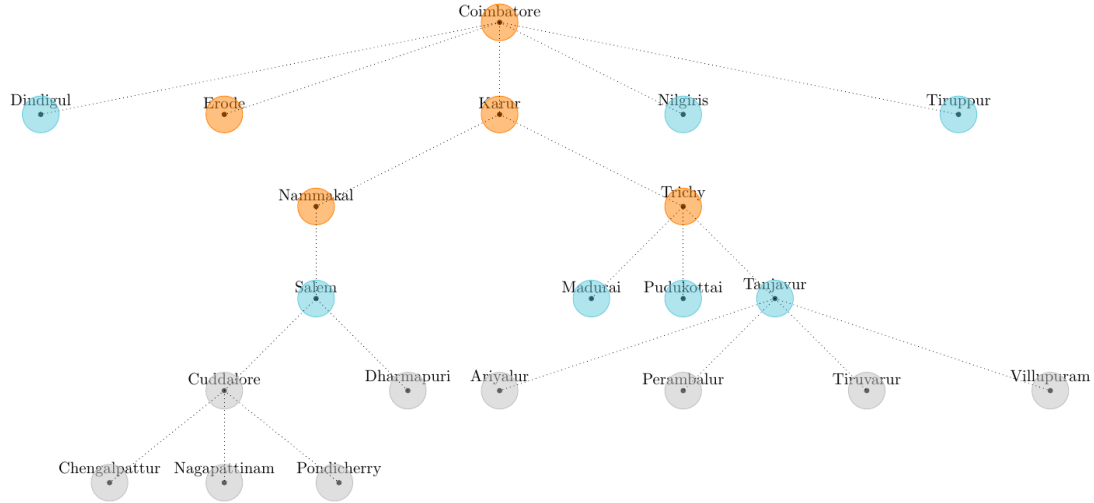


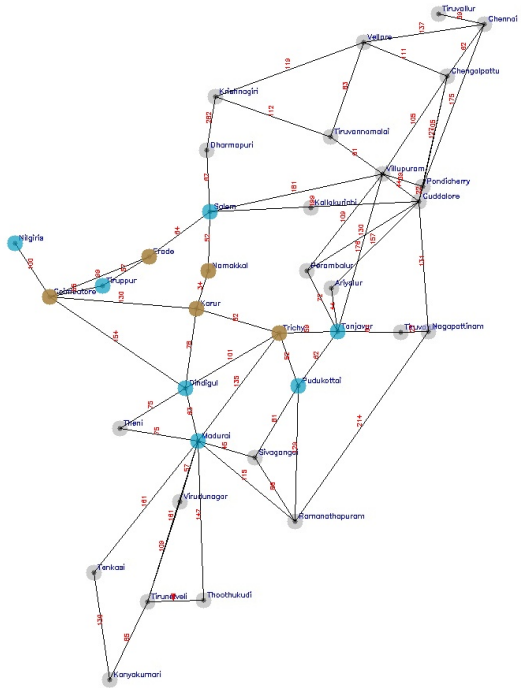
Fontier: ['Dindigul' - 549,'Nilgiris' - 621,'Tiruppur' - 480,'Trichy' - 472,'Salem' - 487,]



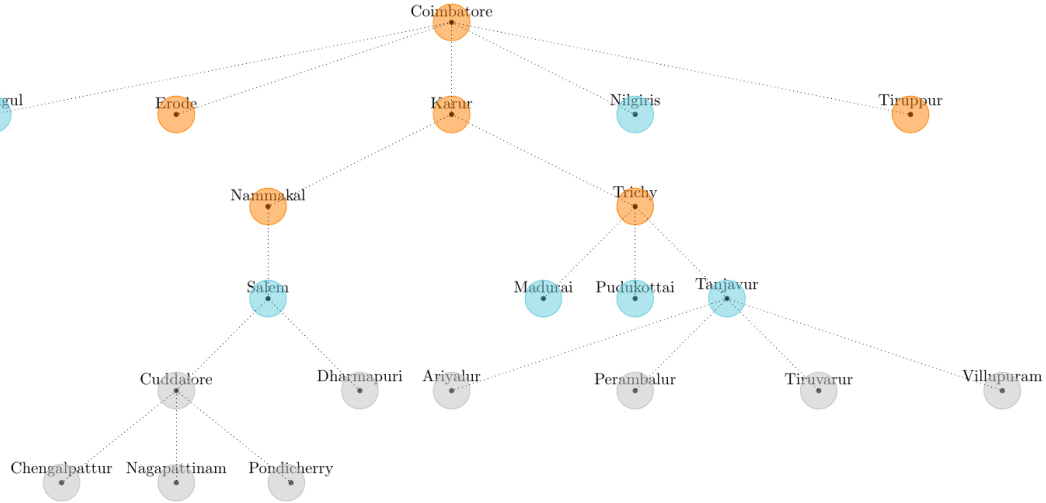


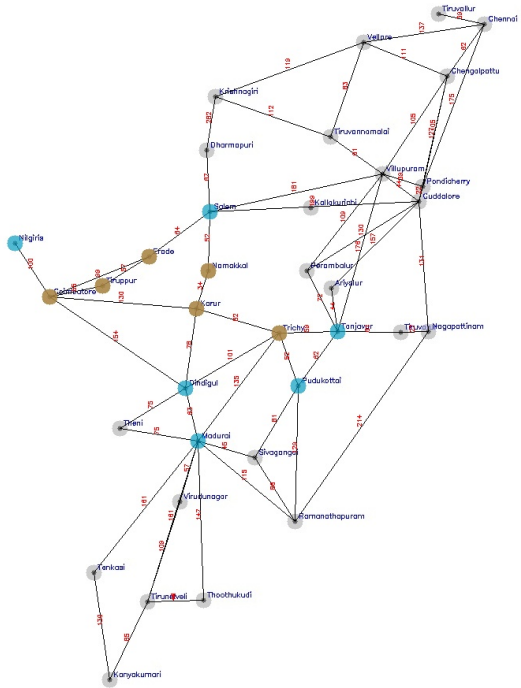
Fontier: ['Dindigul' - 549,'Nilgiris' - 621,'Tiruppur' - 480,'Salem' - 487,'Madurai' - 777,'Pudukottai' - 562,'Tanjavur' - 484,]



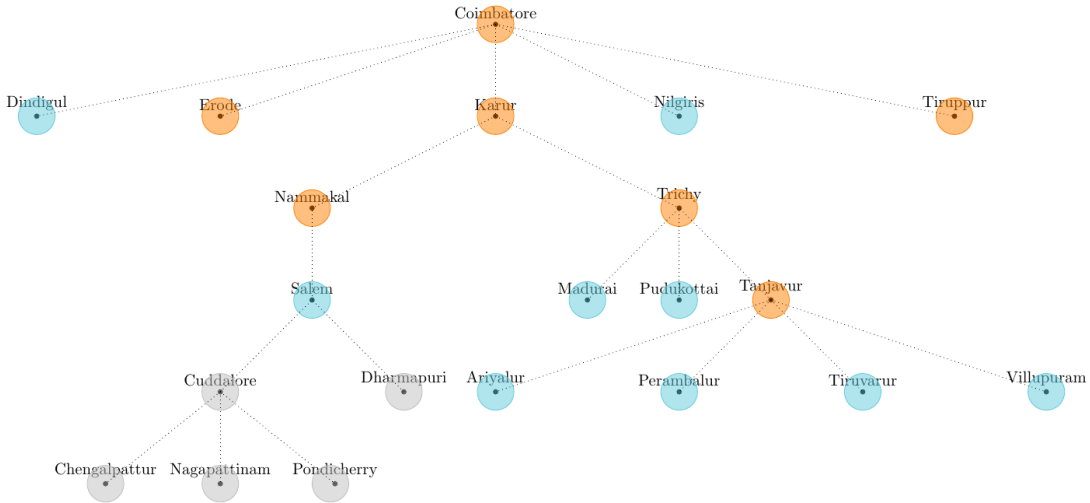


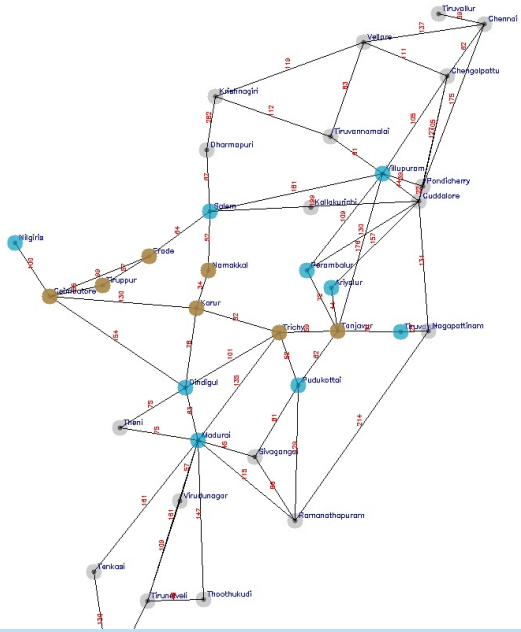
Fontier: ['Dindigul' - 549,'Nilgiris' - 621,'Salem' - 487,'Madurai' - 777,'Pudukottai' - 562,'Tanjavur' - 484,]



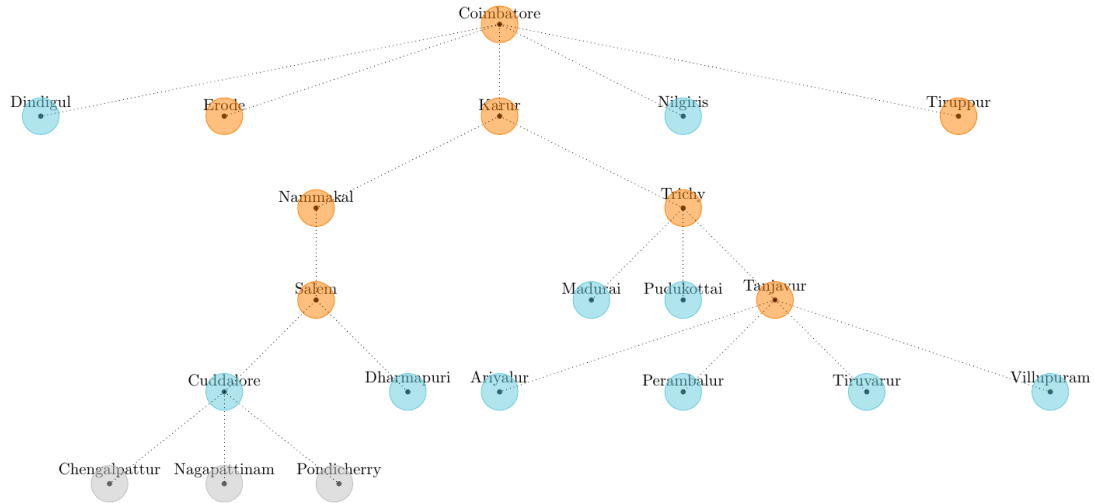


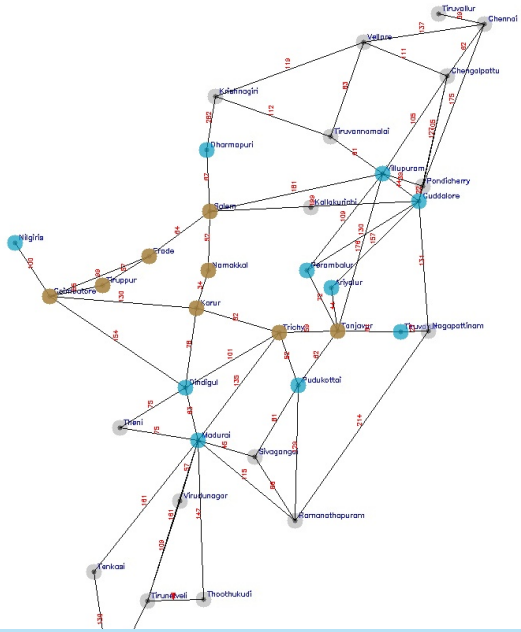
Fontier: ['Dindigul' - 549,'Nilgiris' - 621,'Salem' - 487,'Madurai' - 777,'Pudukottai' - 562,'Ariyalur' - 488,'Perambalur' - 524,'Tiruvarur' - 519,'Villupuram' - 500,]



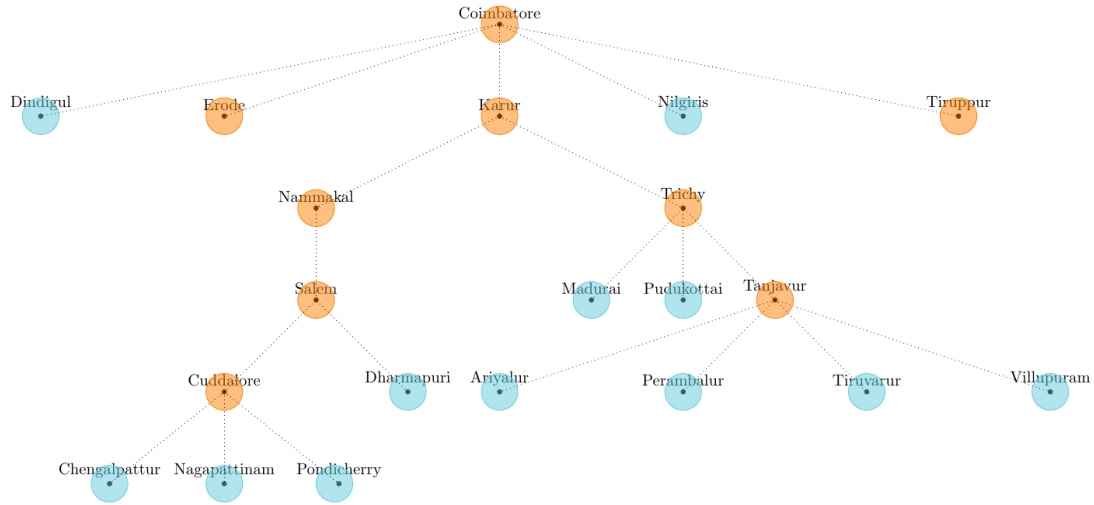


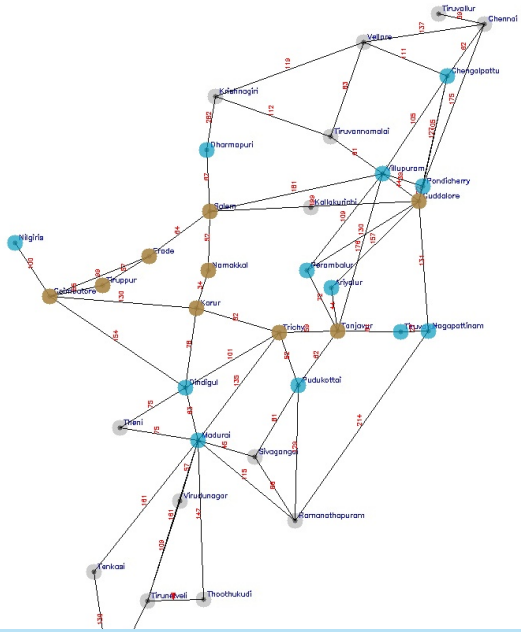
Fontier: ['Dindigul' - 549,'Nilgiris' - 621,'Madurai' - 777,'Pudukottai' - 562,'Ariyalur' - 488,'Perambalur' - 524,'Tiruvarur' - 519,'Villupuram' - 500,'Cuddalore' - 434,'Dharmapuri' - 560,]



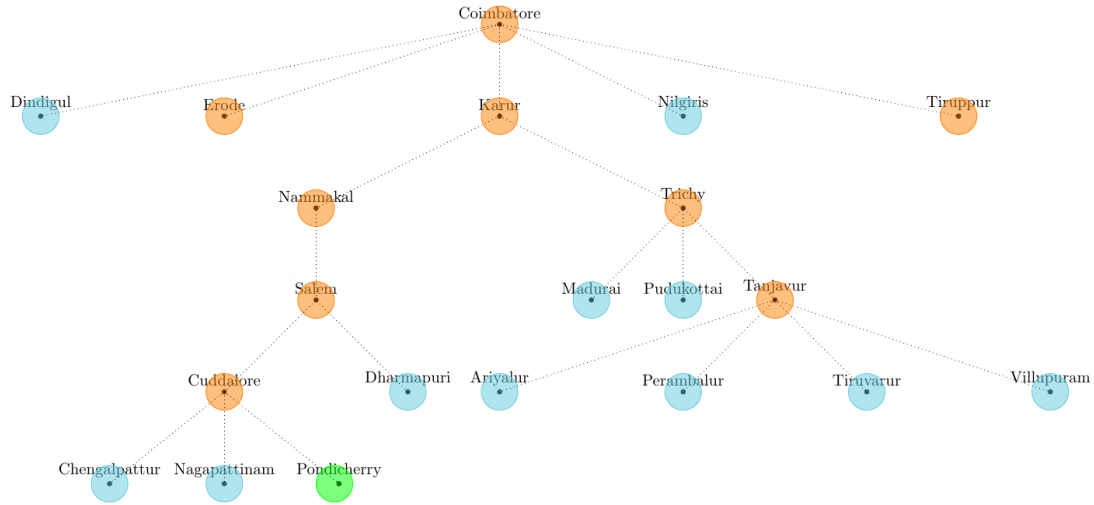


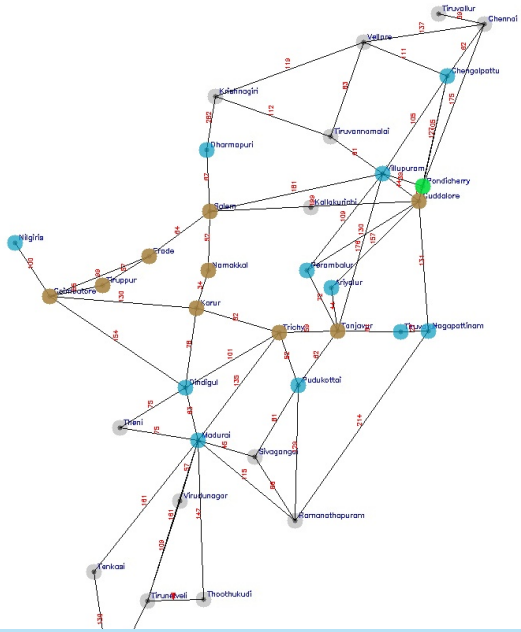
Fontier: ['Dindigul' - 549,'Nilgiris' - 621,'Madurai' - 777,'Pudukottai' - 562,'Ariyalur' - 488,'Perambalur' - 524,'Tiruvarur' - 519,'Villupuram' - 500,'Dharmapuri' - 560,'Chengalpattu' - 685,'Nagapattinam' - 730,'Pondicherry' - 437,]



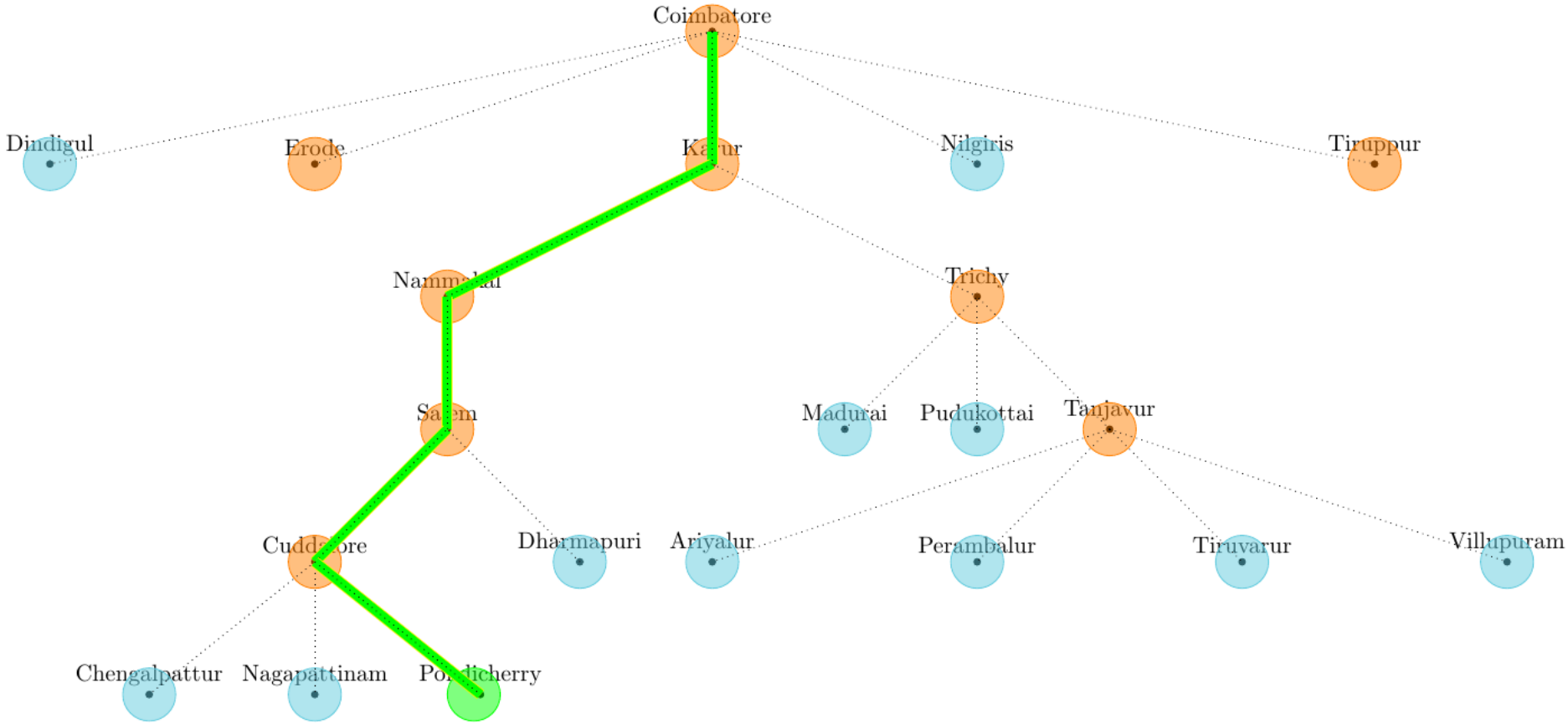


Fontier: ['Dindigul' - 549,'Nilgiris' - 621,'Madurai' - 777,'Pudukottai' - 562,'Ariyalur' - 488,'Perambalur' - 524,'Tiruvarur' - 519,'Villupuram' - 500,'Dharmapuri' - 560,'Chengalpattu' - 685,'Nagapattinam' - 730,]



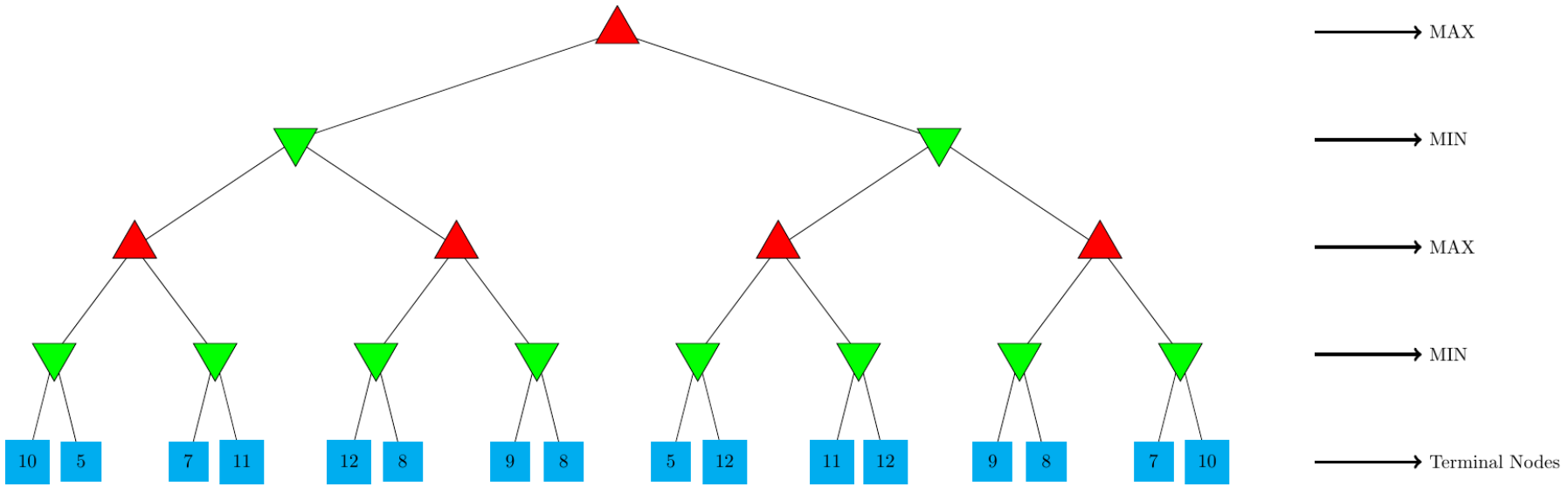


The path A^* took is from Coimbatore to Pondicherry is Coimbatore $->$ Karur $->$ Namakkal $->$ Salem $->$ Cuddalore $->$ Pondicherry.

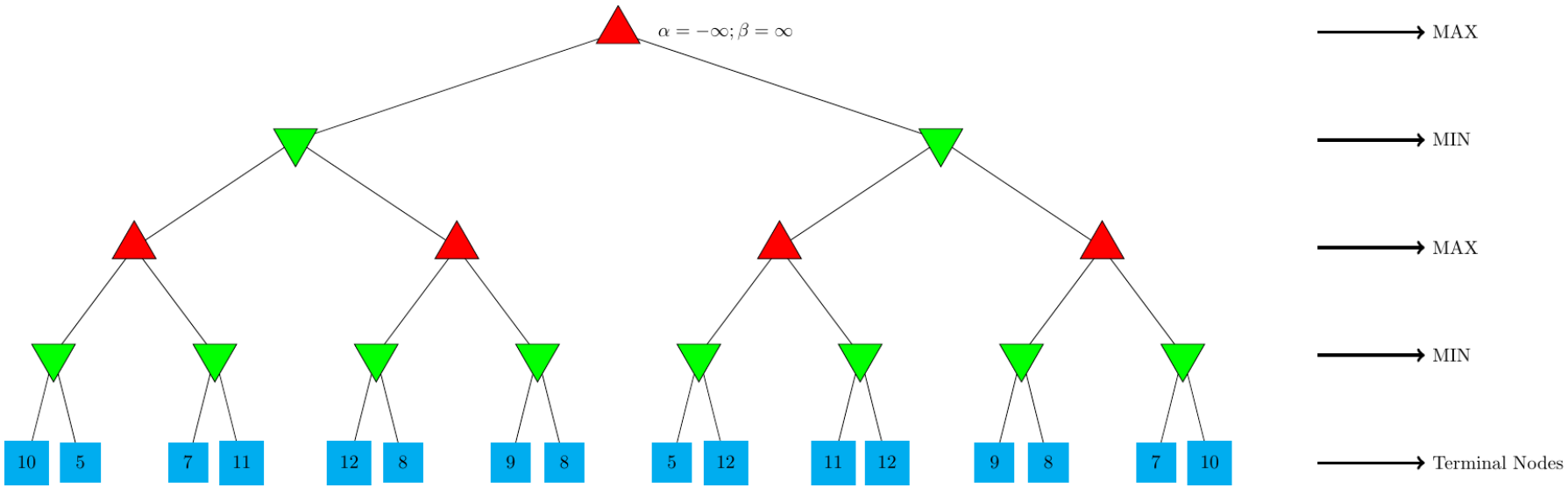


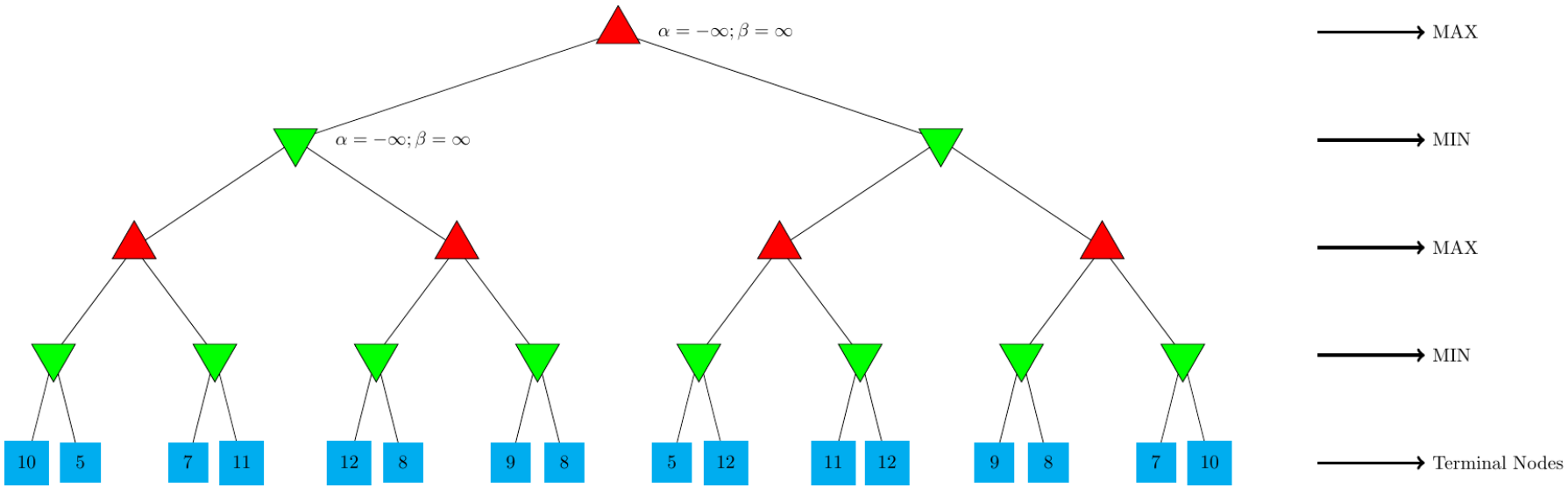
Adverserial Search - Alpha Beta In Search Tree

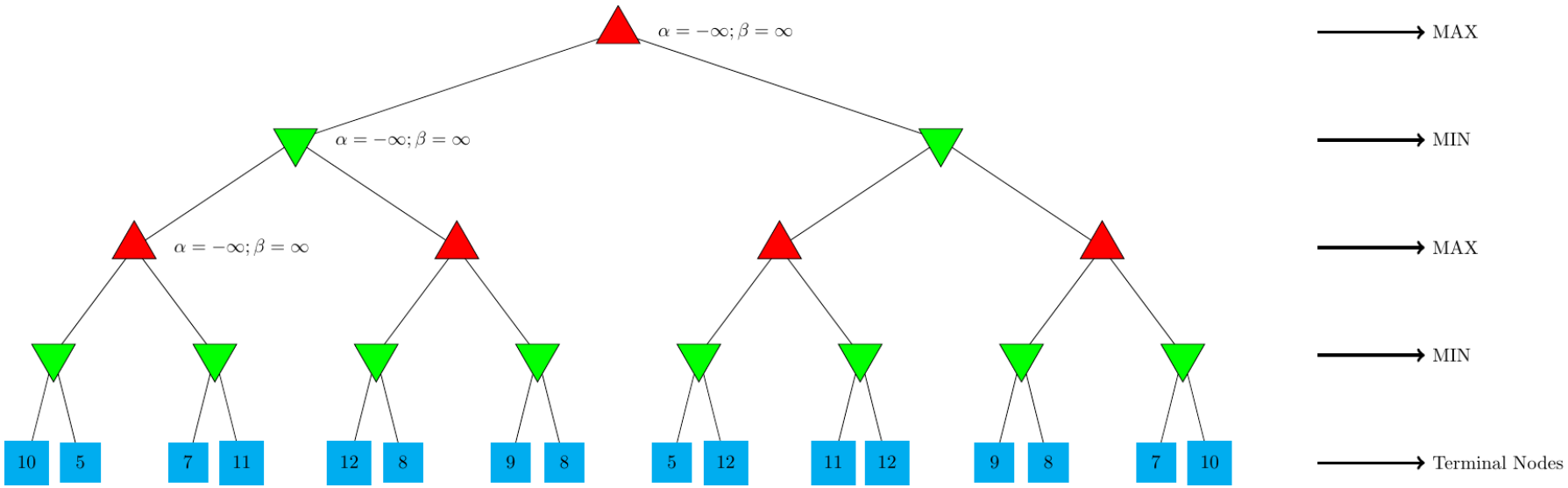
- The Search Tree is as below

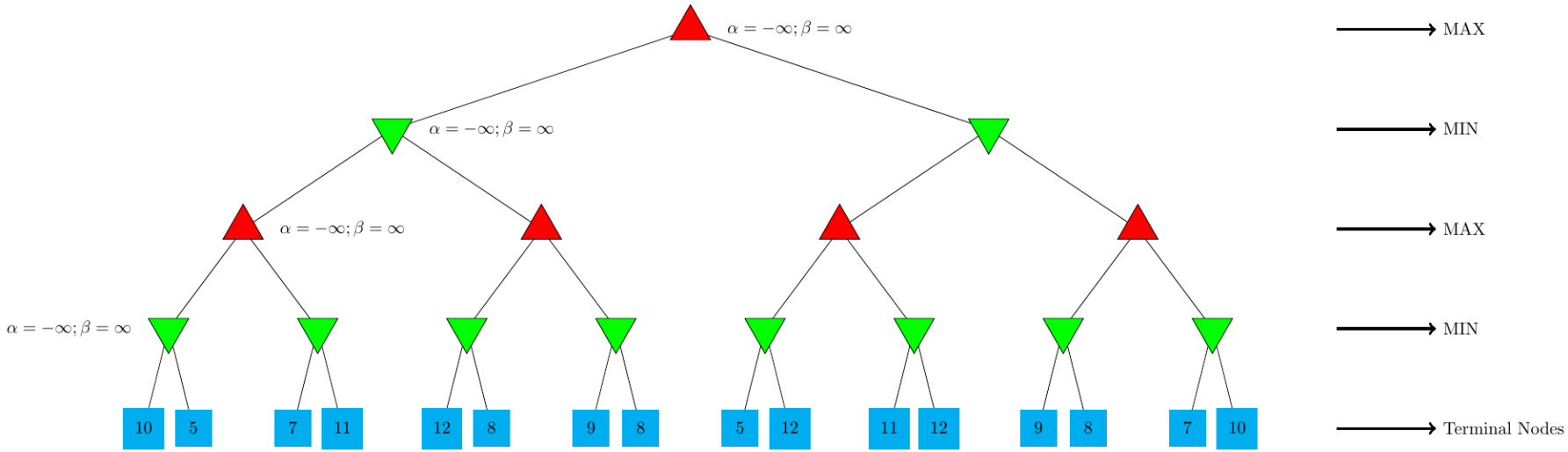


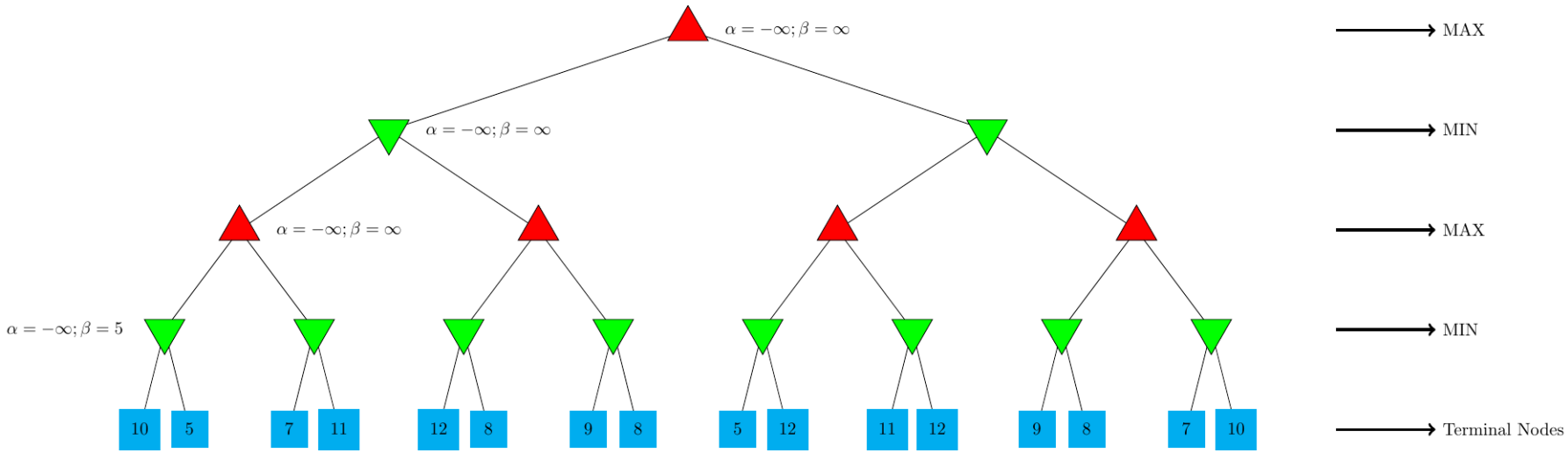
- When $\alpha \geq \beta$, we prune.

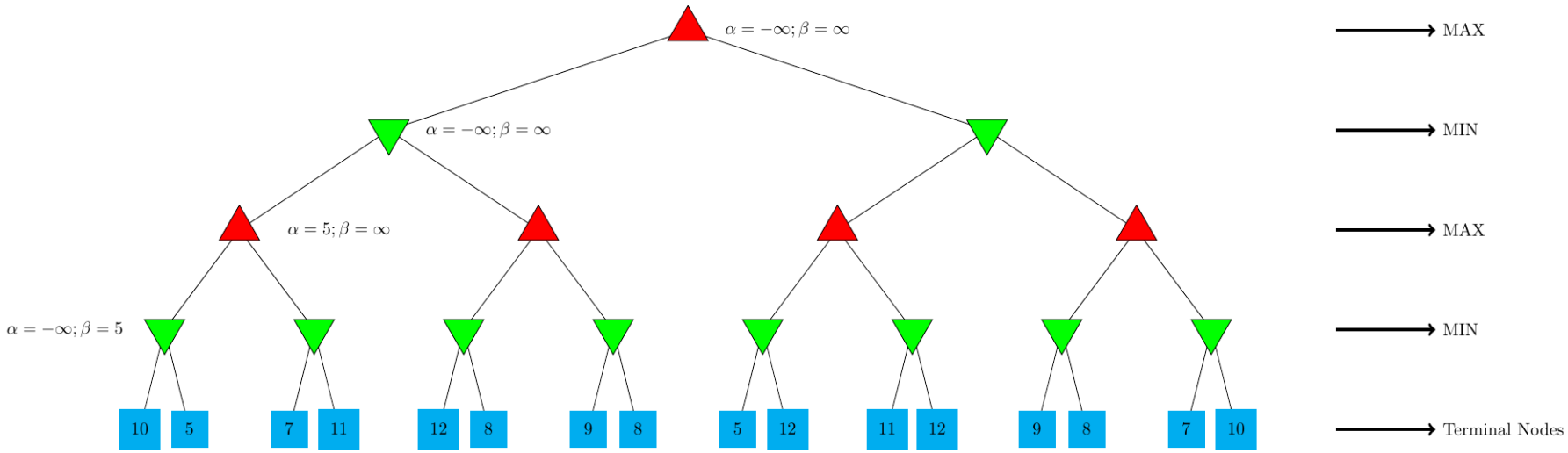

$\alpha = -\infty; \beta = \infty$
MAX
MIN
MAX
MIN
Terminal Nodes
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10

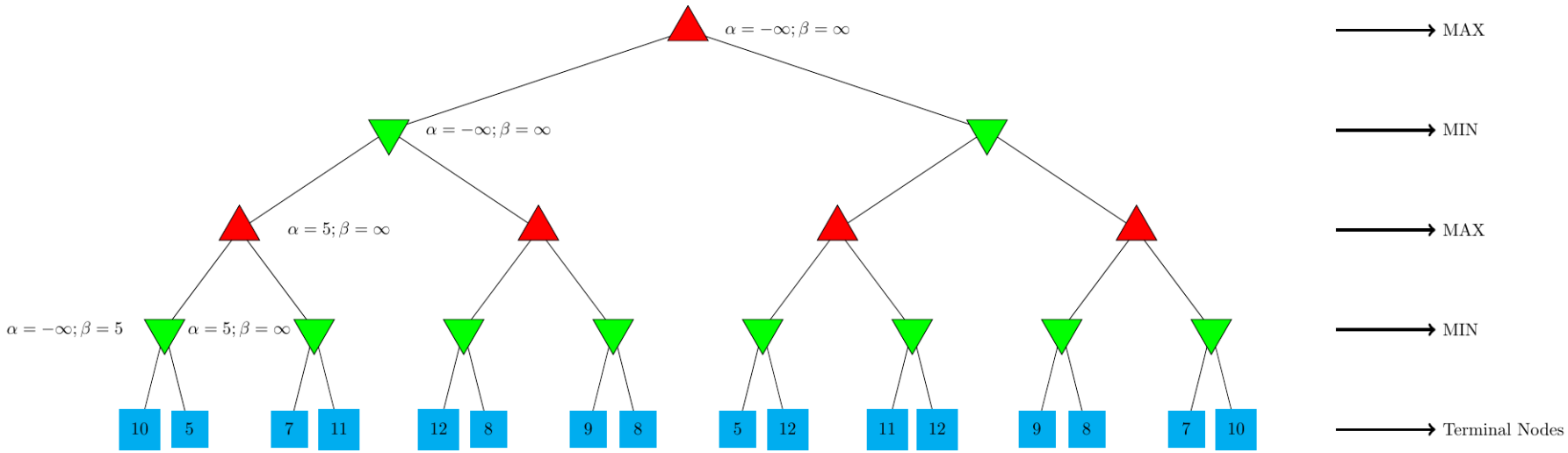

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

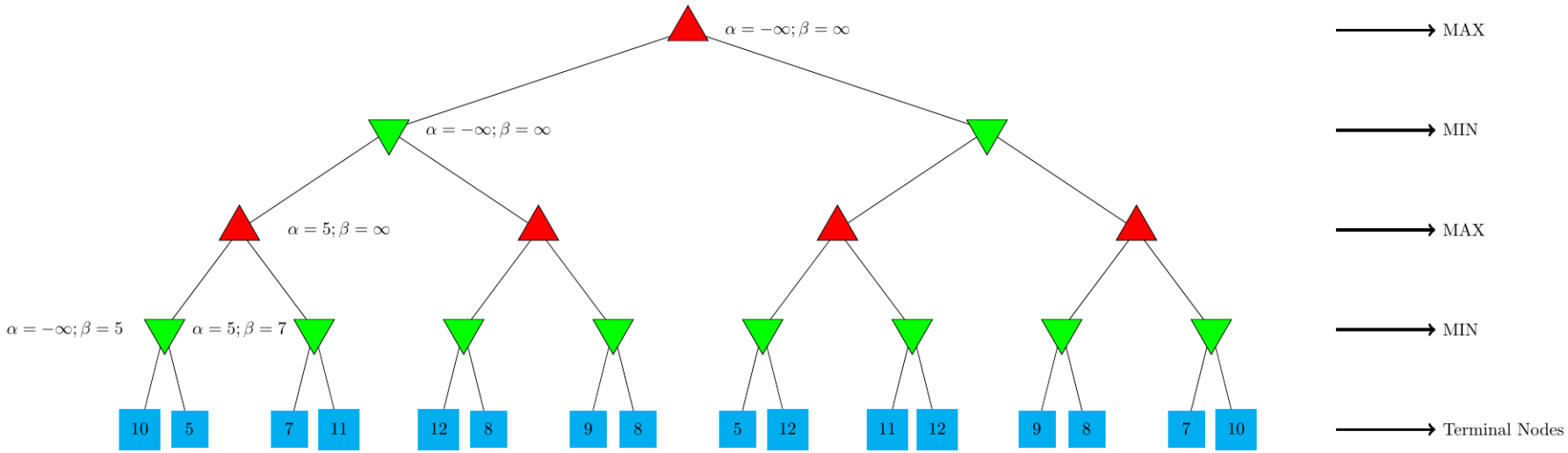

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

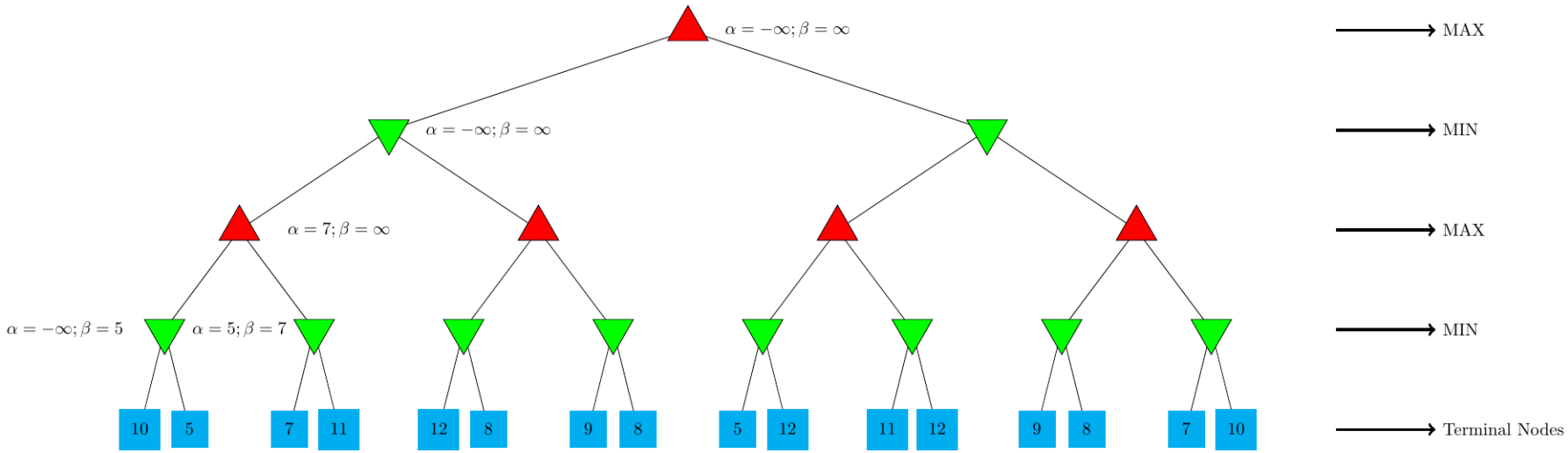

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

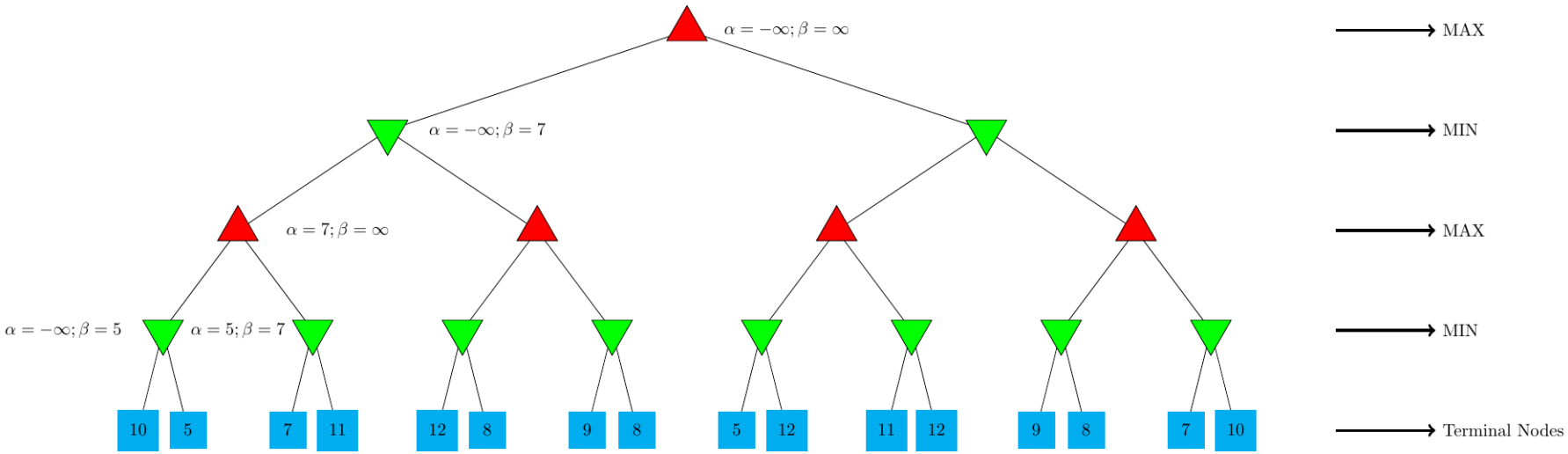

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = 5$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

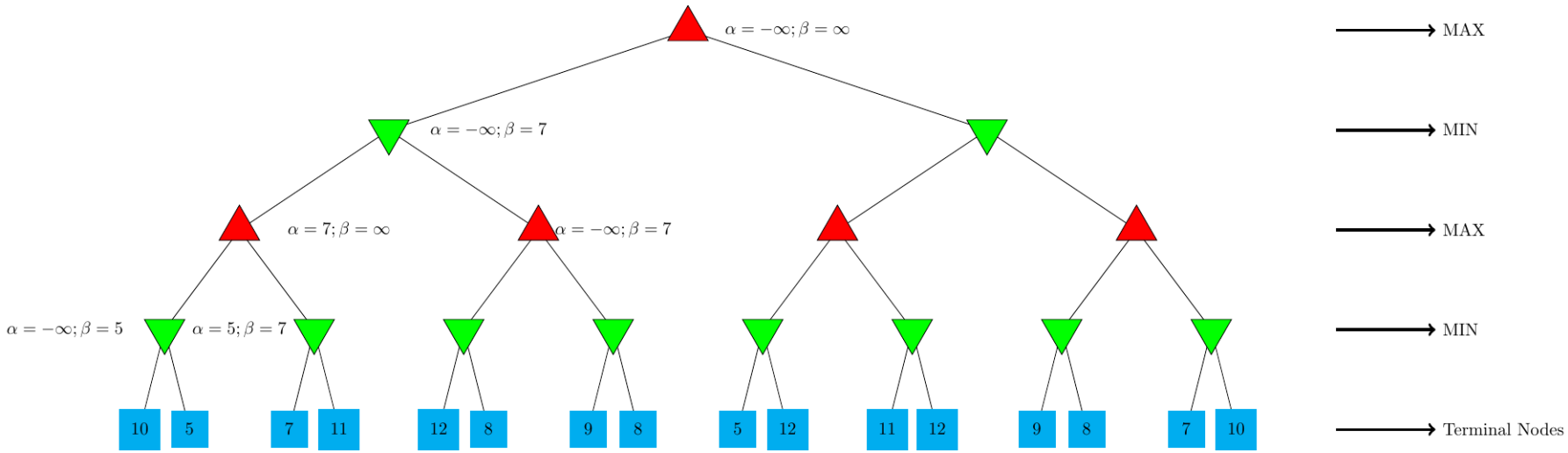

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
$\alpha = 5; \beta = \infty$
$\alpha = -\infty; \beta = 5$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

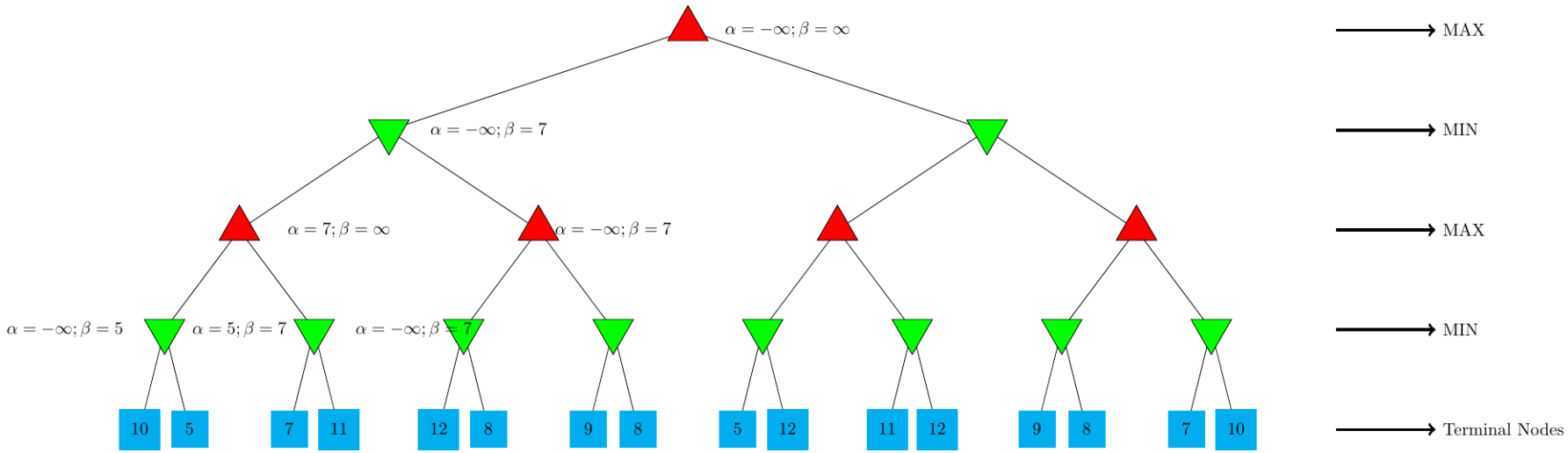

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
$\alpha = 5; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = \infty$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

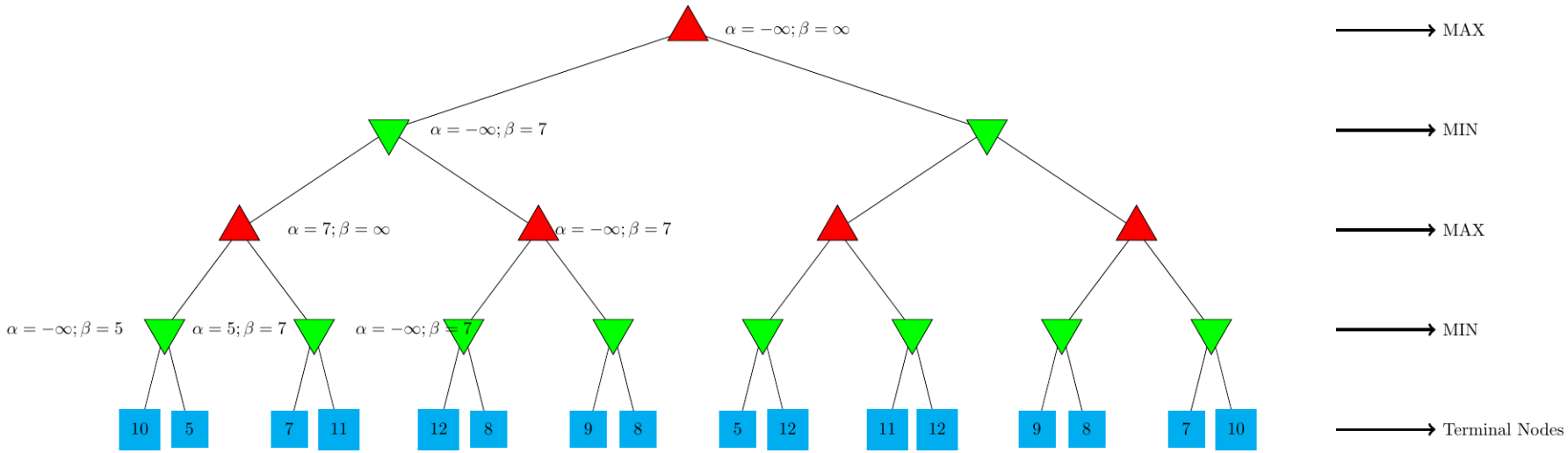

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
$\alpha = 5; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

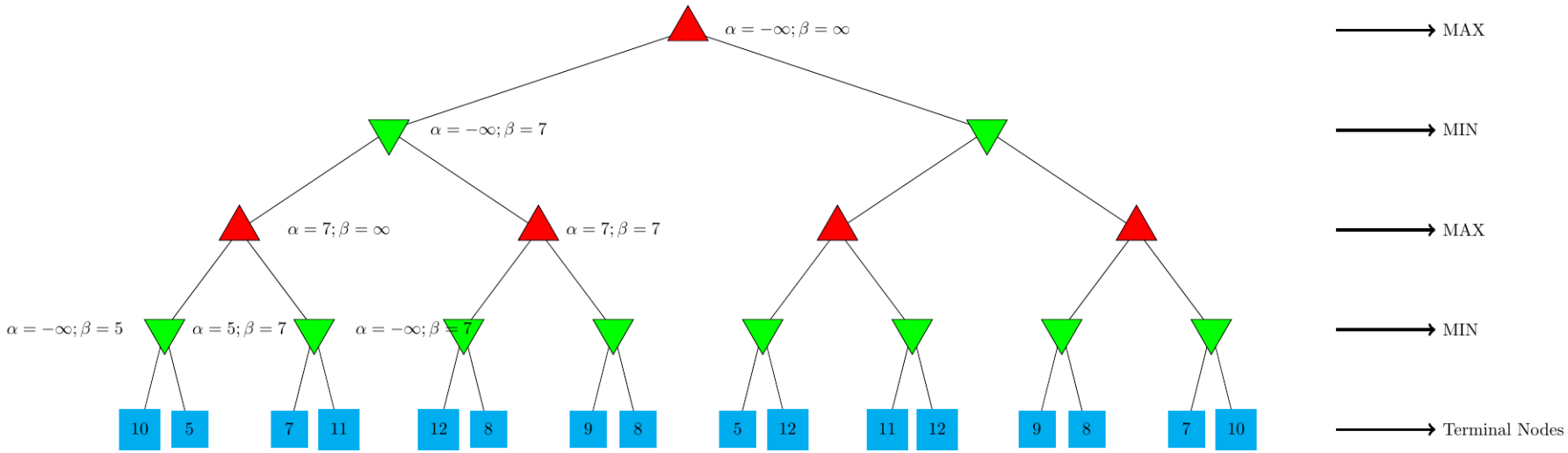

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = \infty$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes


$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

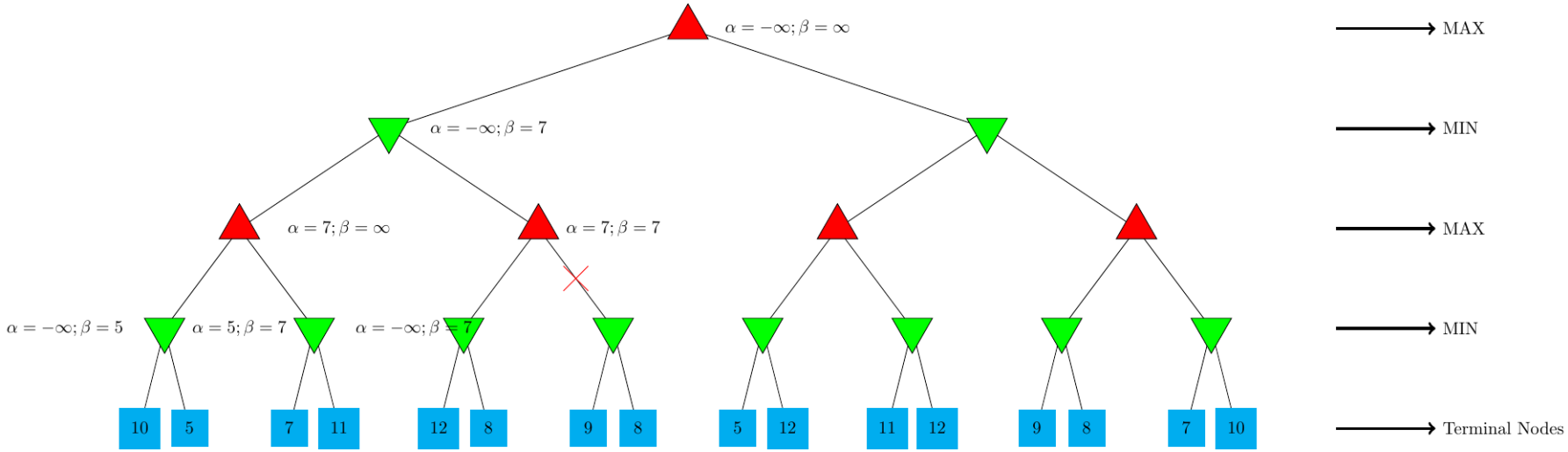

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

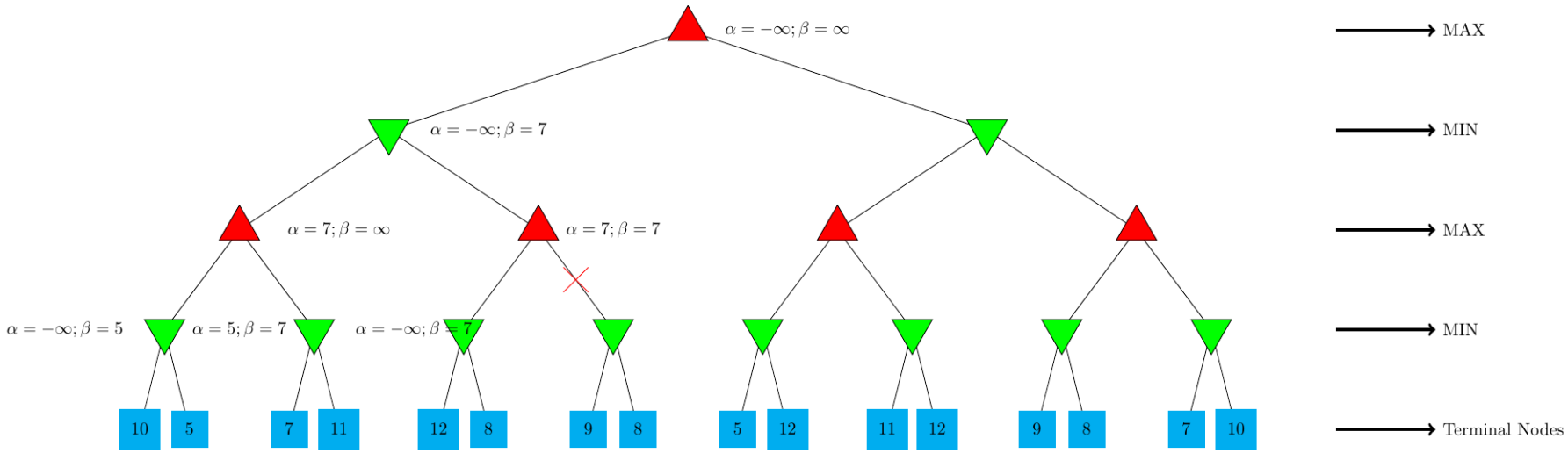

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

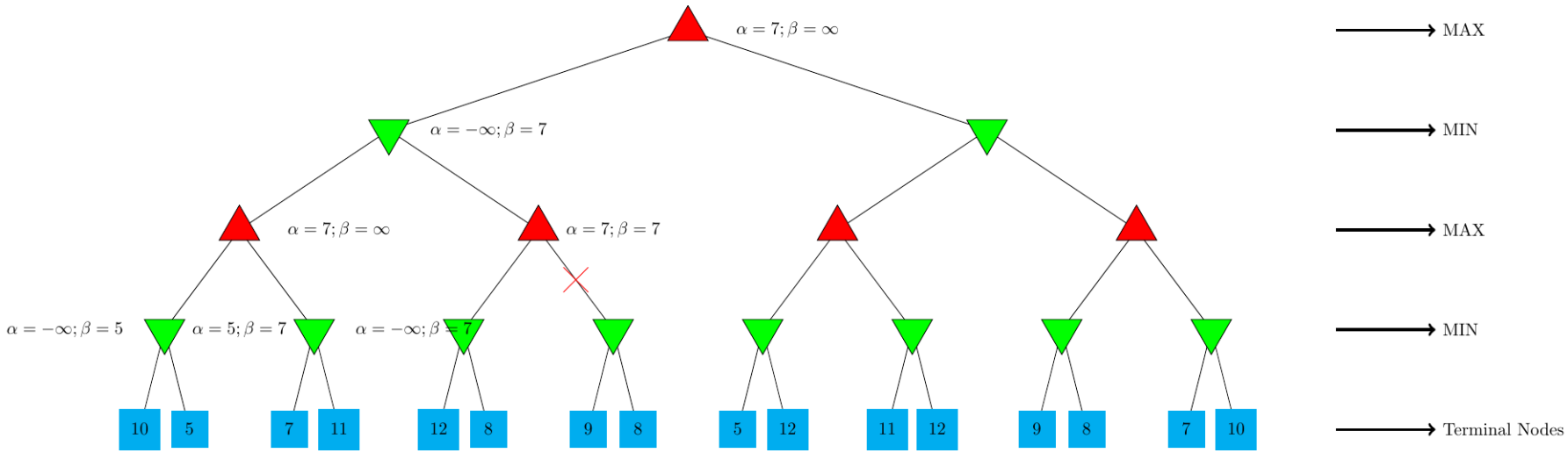

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

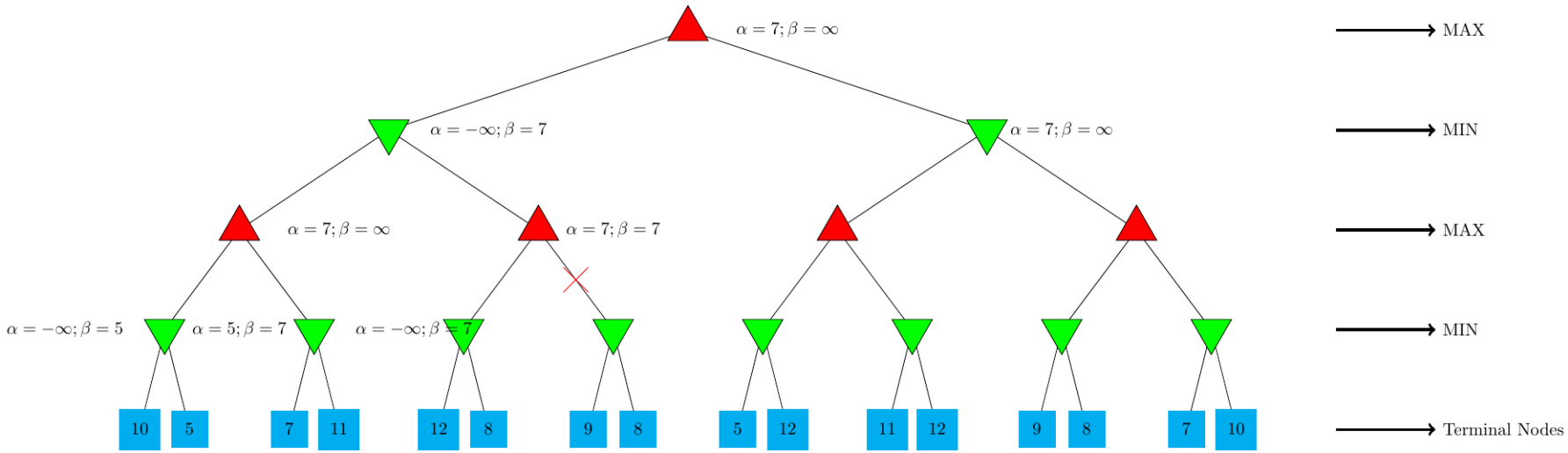

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

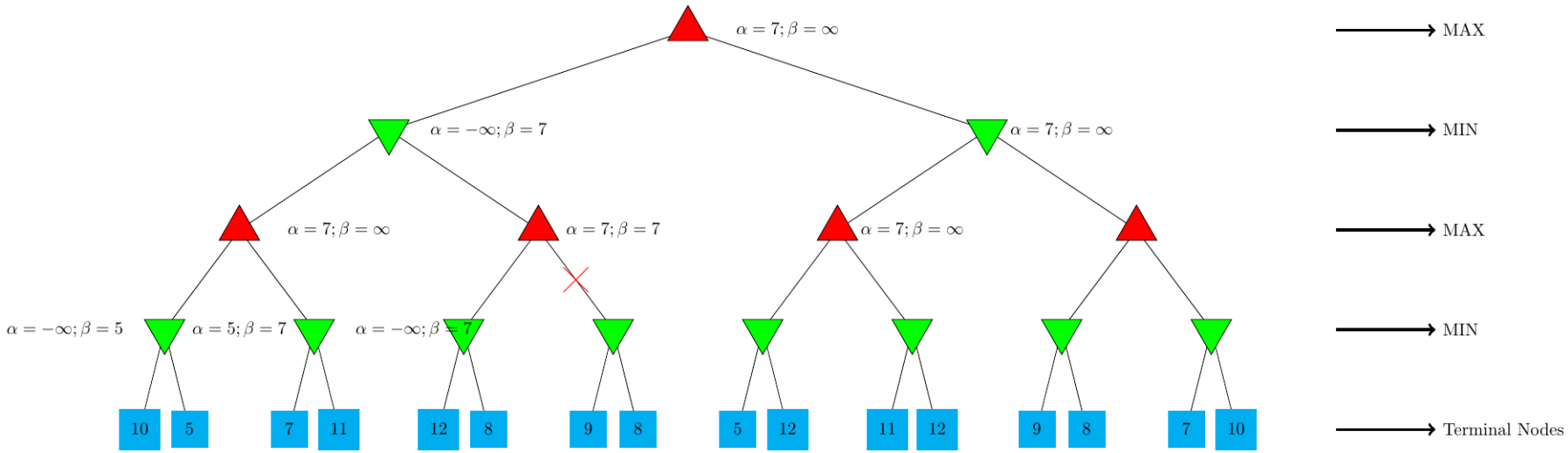
Here, $\alpha \geq \beta$, so we prune.

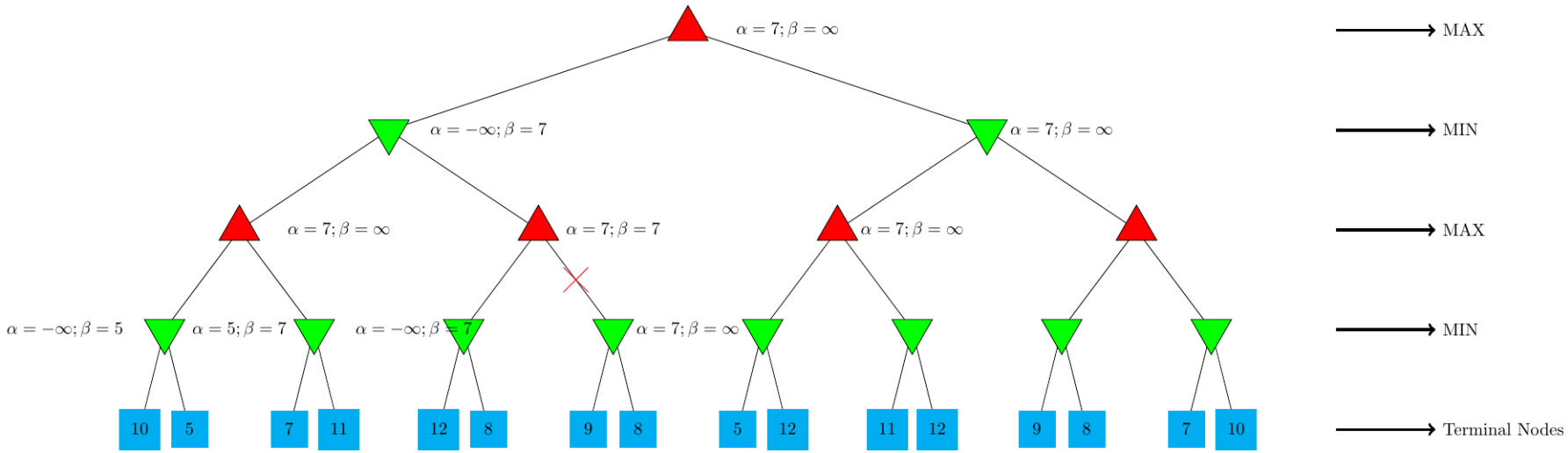


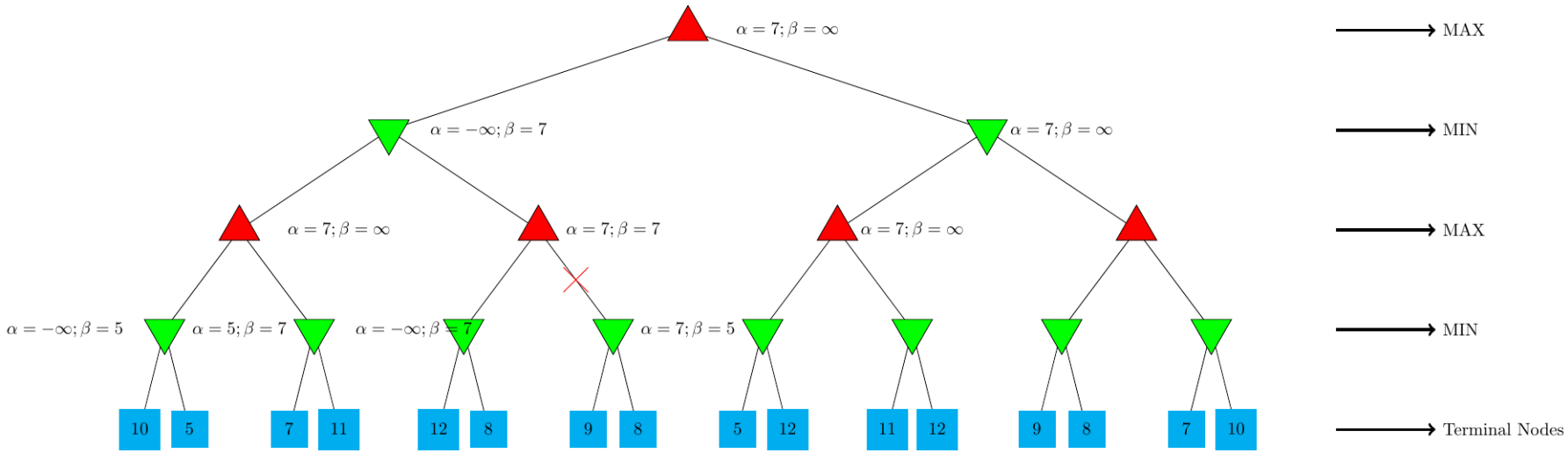

$\alpha = -\infty; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes


$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

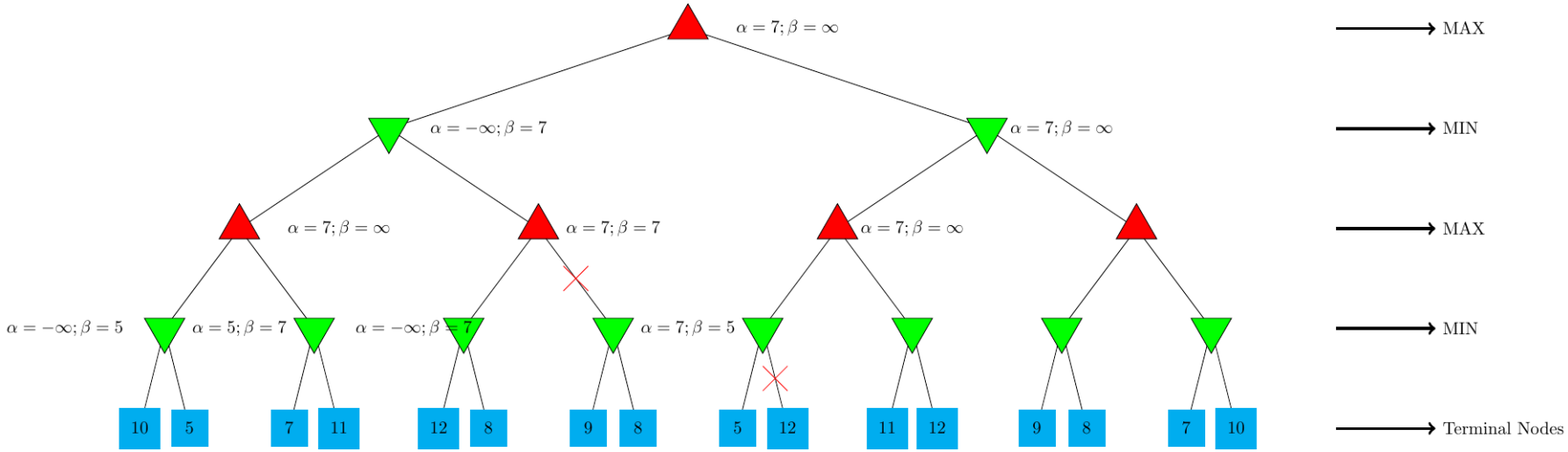

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

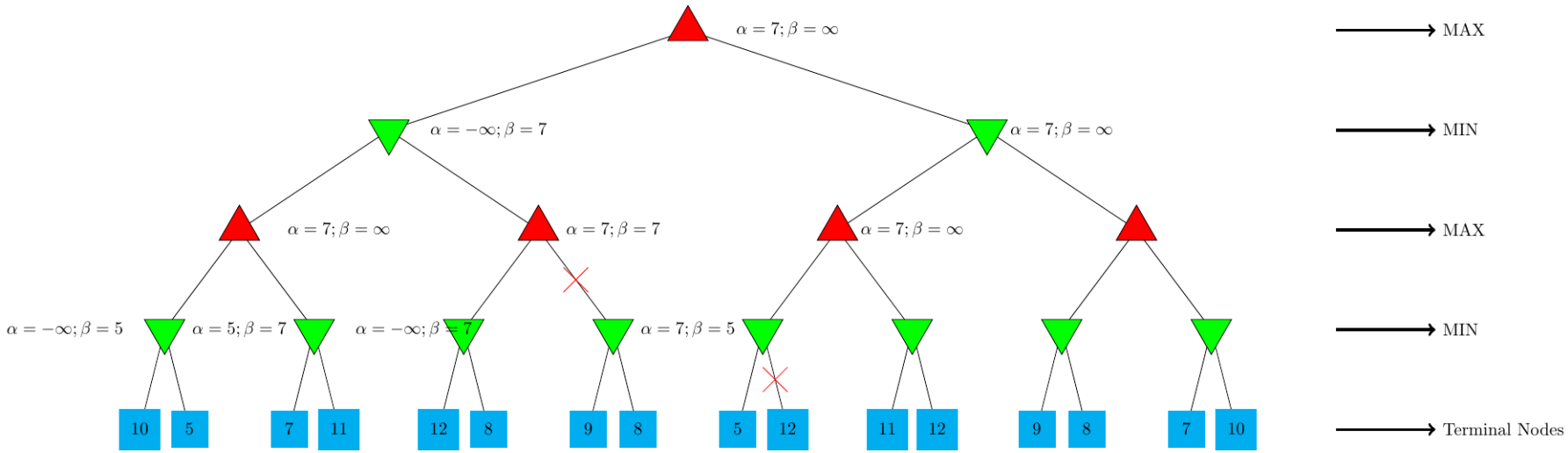

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

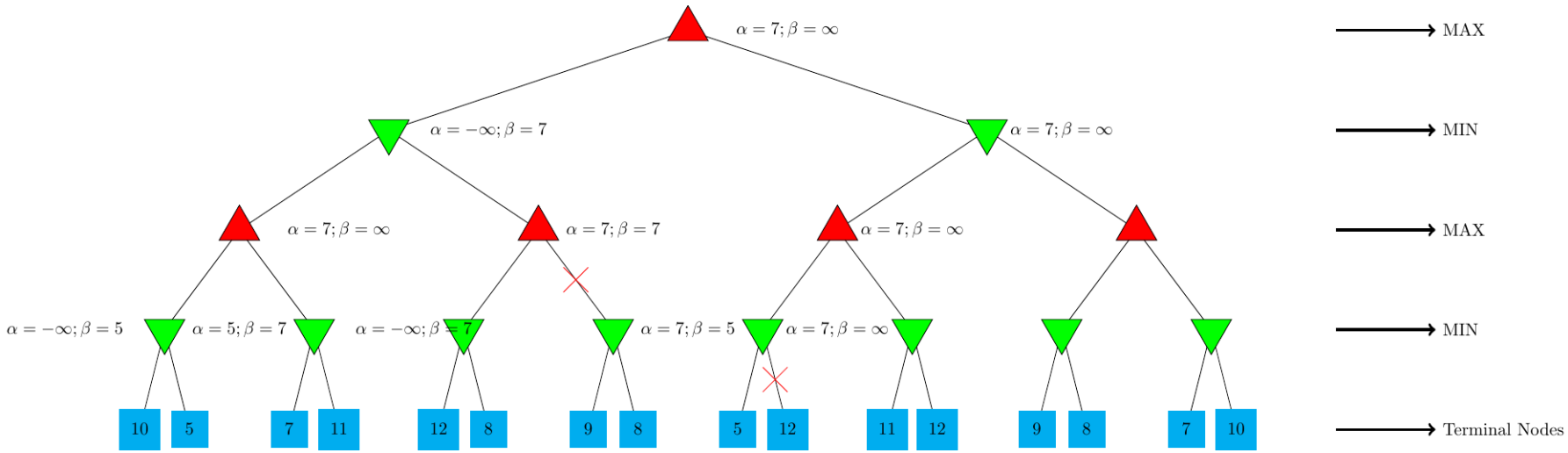

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

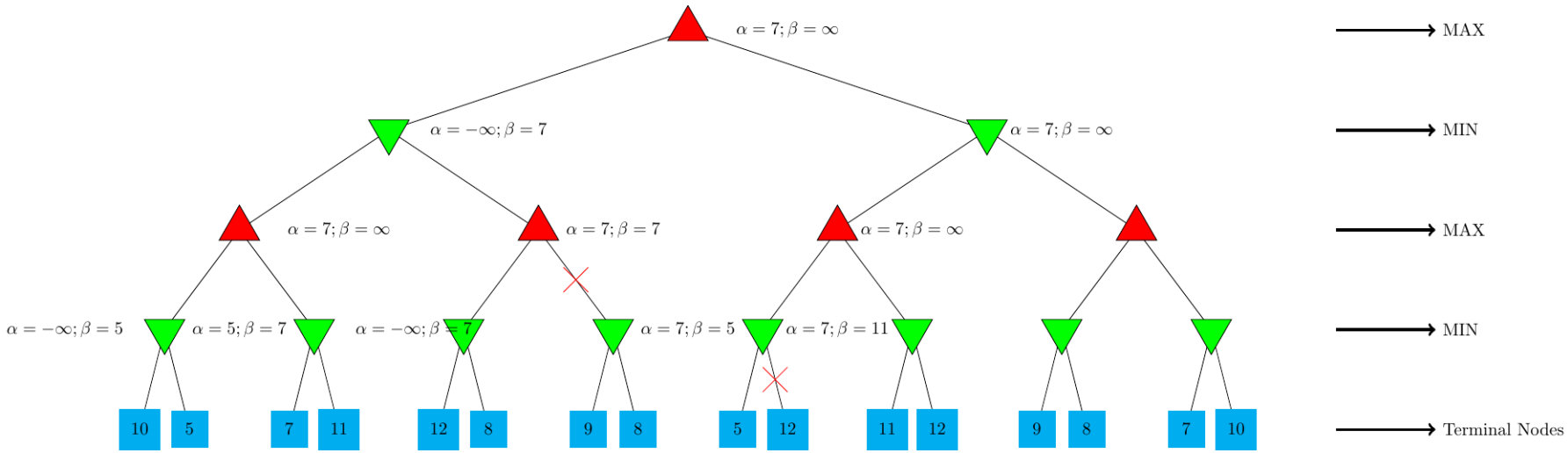

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

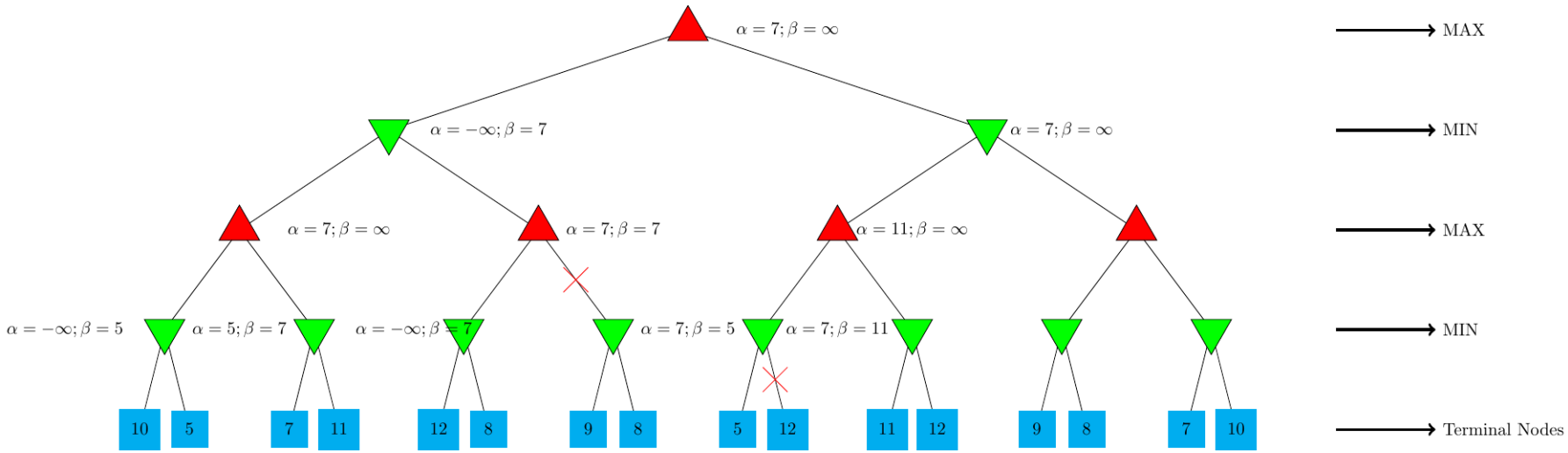
Here, $\alpha \geq \beta$, so we prune.

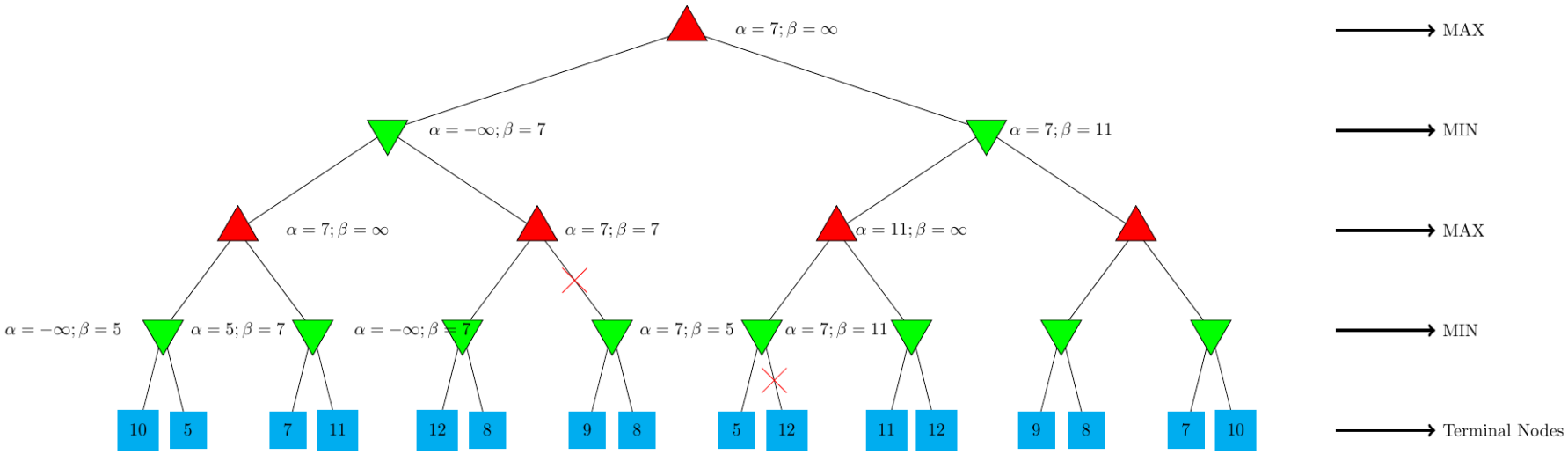


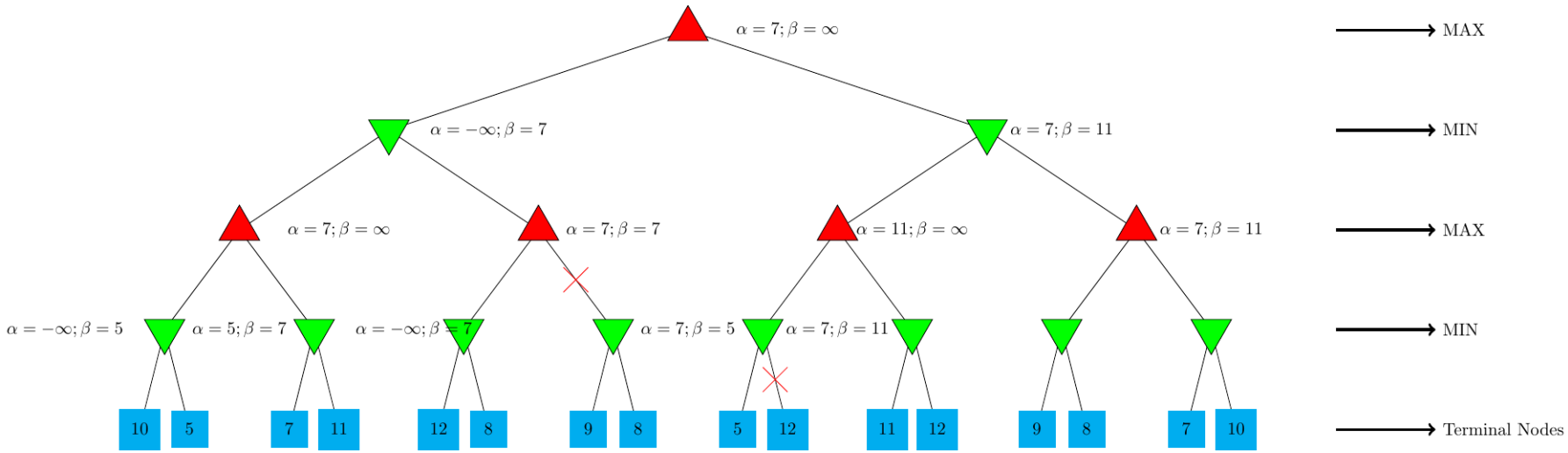

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

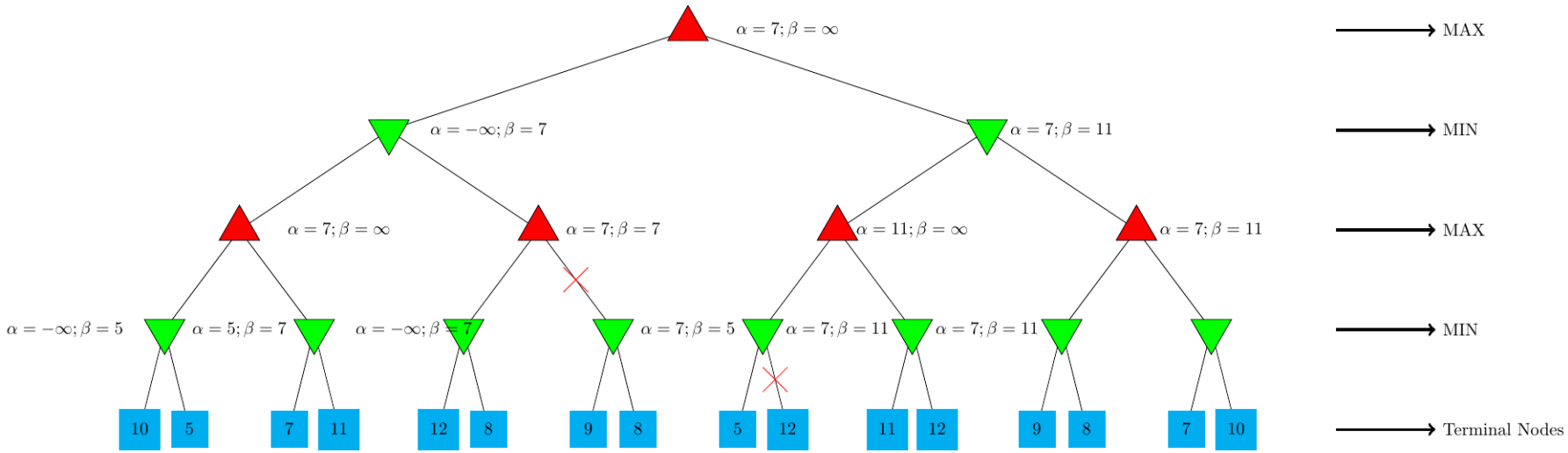

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = \infty$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

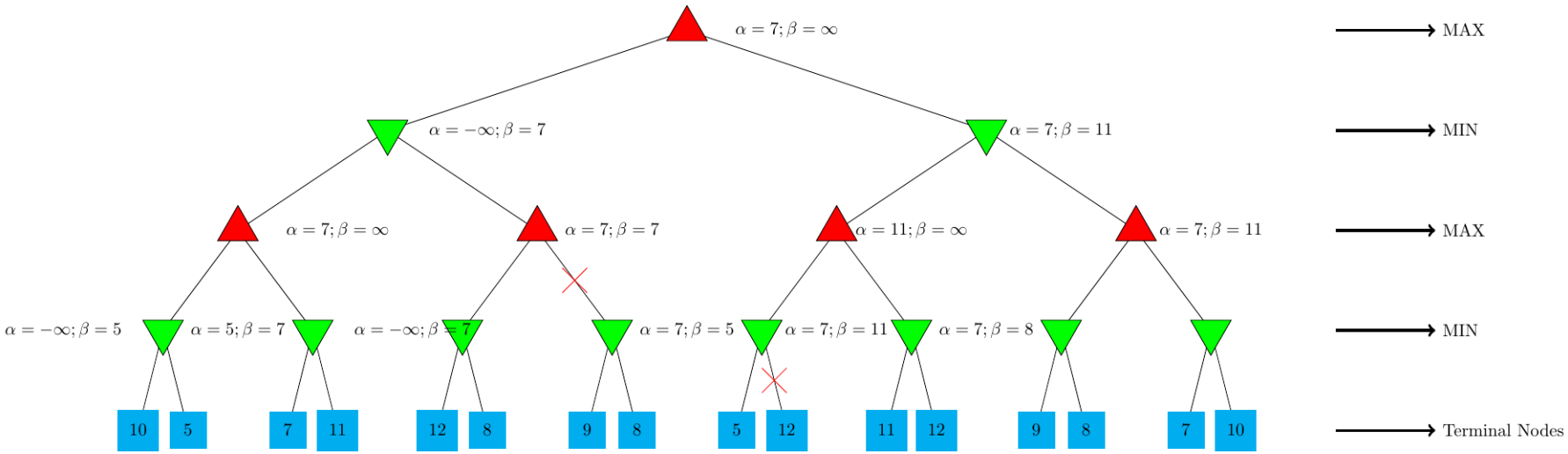

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

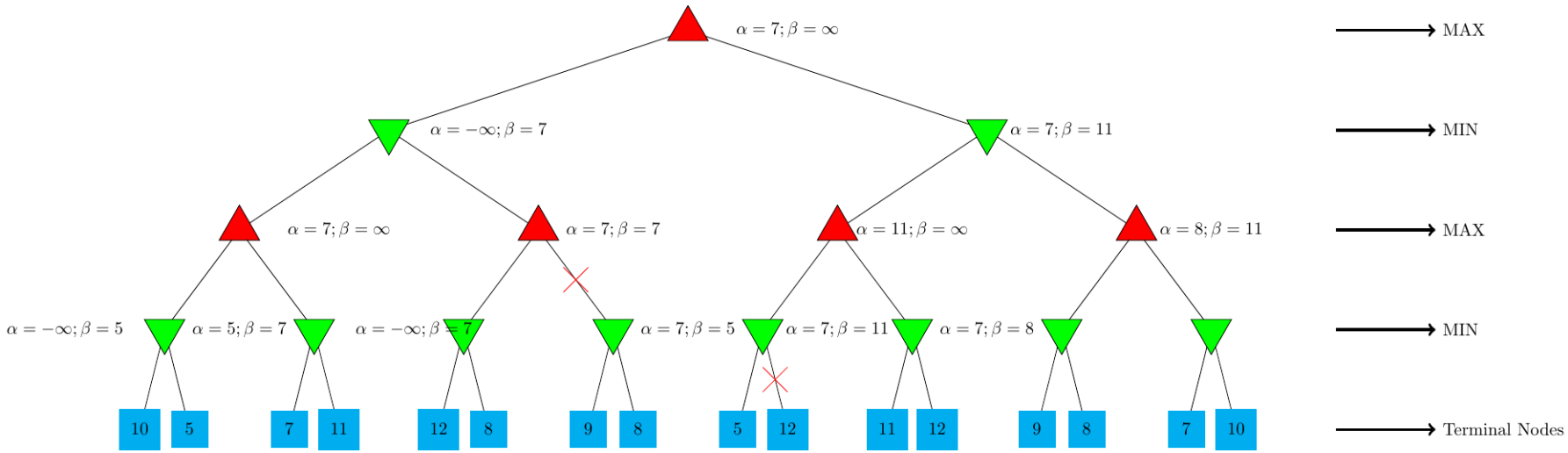

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

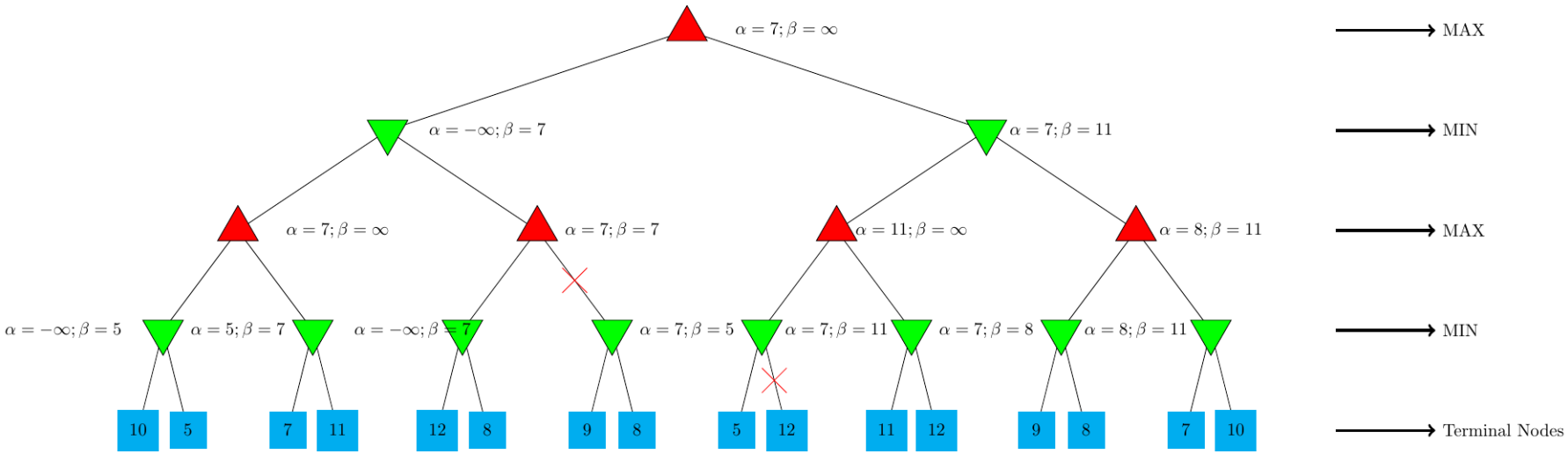

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

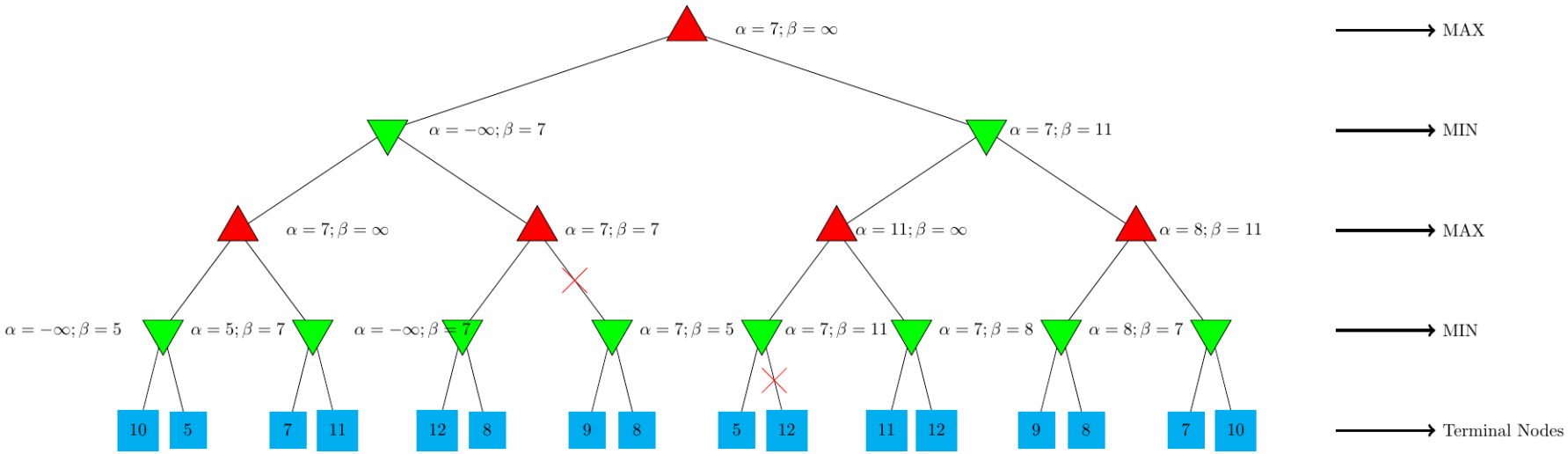

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = 7; \beta = 11$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes


$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = 7; \beta = 11$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = 11$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

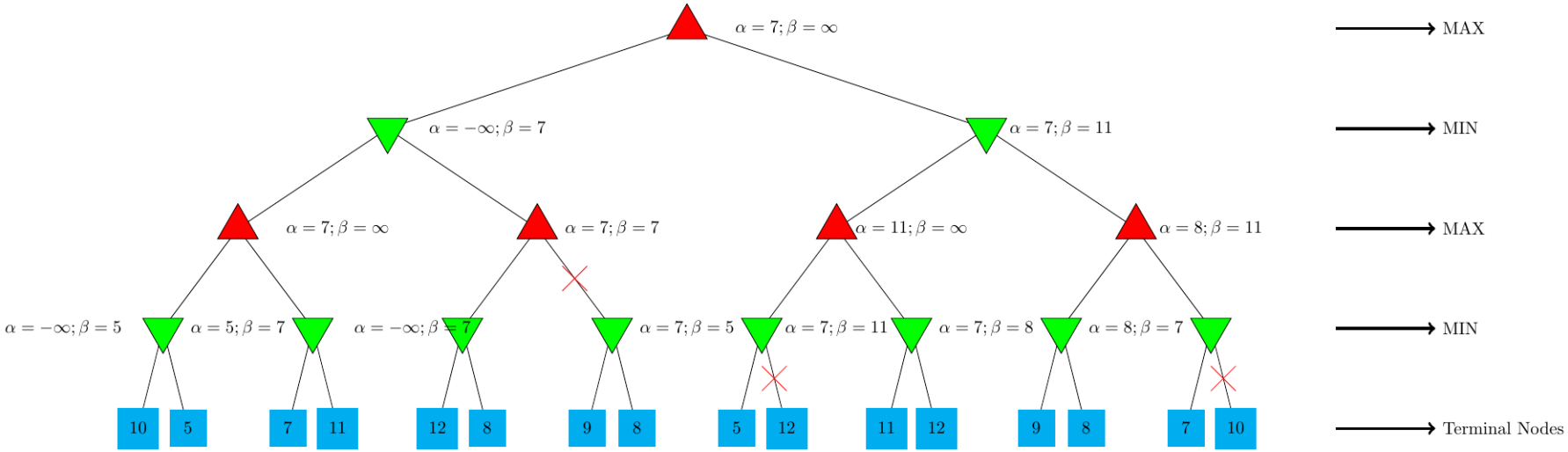

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = 7; \beta = 11$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = 8$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

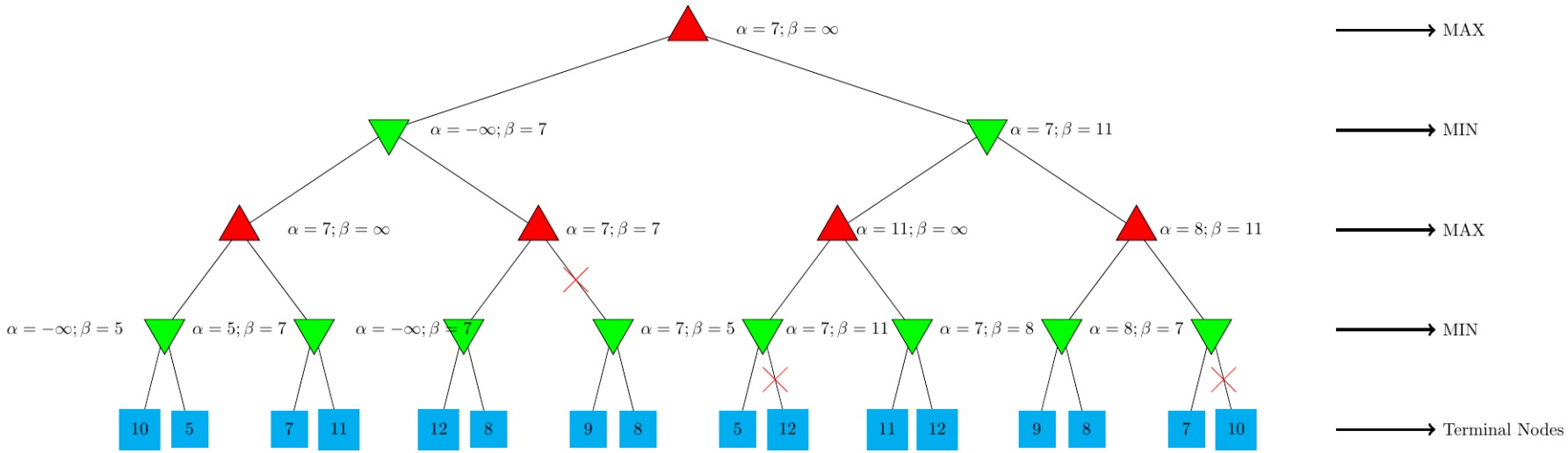

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = 8; \beta = 11$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = 8$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

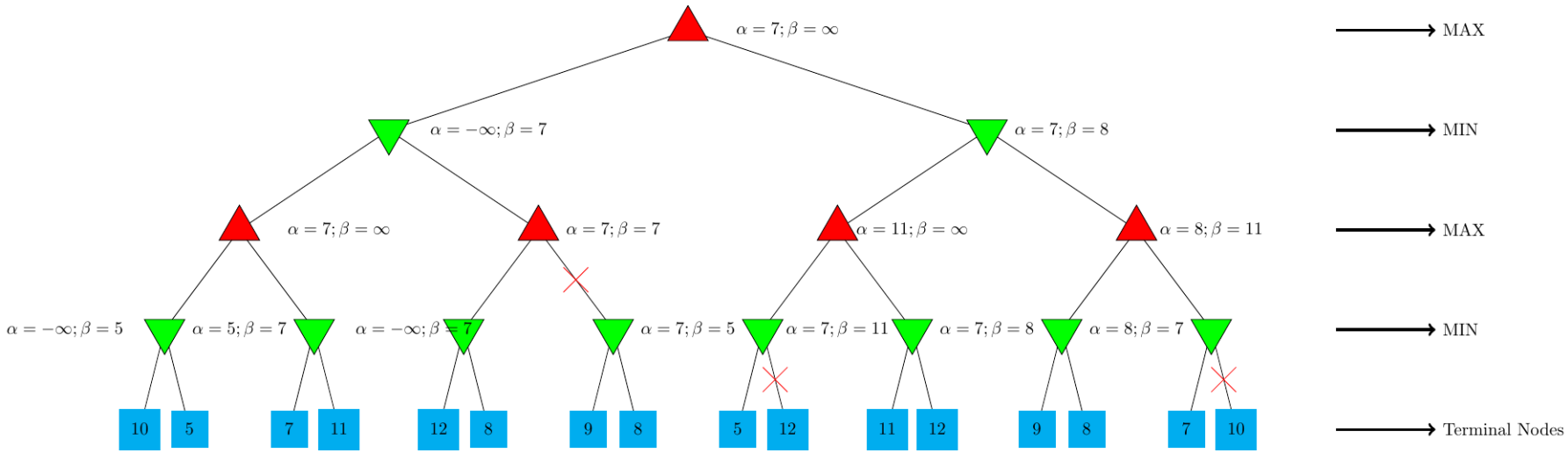

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = 8; \beta = 11$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = 8$
$\alpha = 8; \beta = 11$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

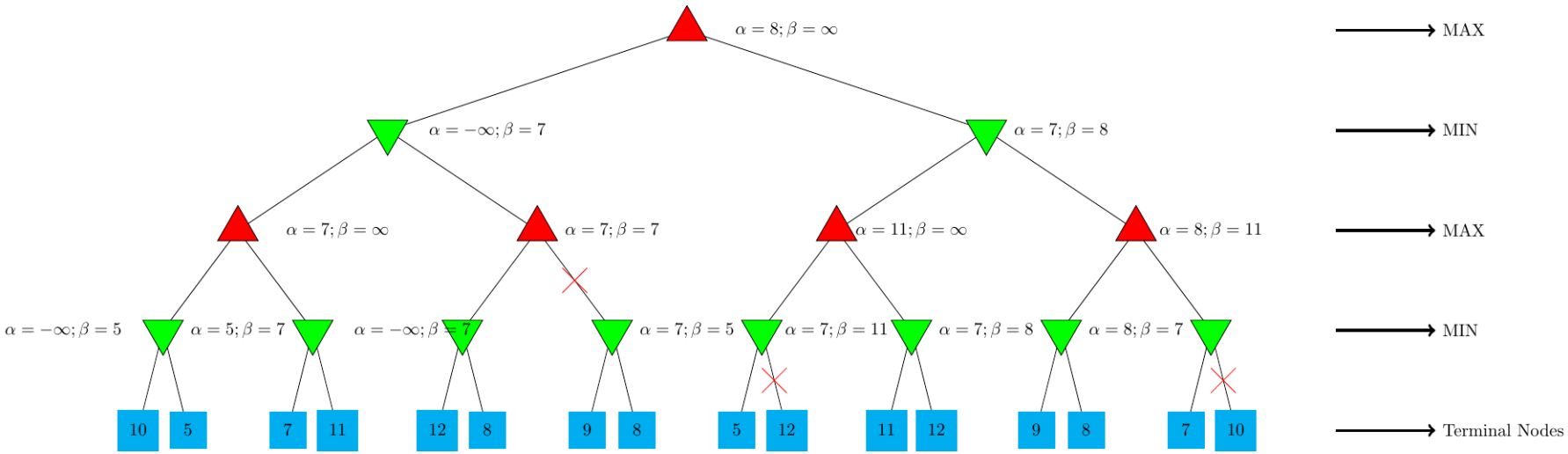

$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = 8; \beta = 11$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = 8$
$\alpha = 8; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

Here, $\alpha \geq \beta$, so we prune.




$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = 8; \beta = 11$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = 8$
$\alpha = 8; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes


$\alpha = 7; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 8$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = 8; \beta = 11$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = 8$
$\alpha = 8; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes


$\alpha = 8; \beta = \infty$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 8$
$\alpha = 7; \beta = \infty$
$\alpha = 7; \beta = 7$
$\alpha = 11; \beta = \infty$
$\alpha = 8; \beta = 11$
$\alpha = -\infty; \beta = 5$
$\alpha = 5; \beta = 7$
$\alpha = -\infty; \beta = 7$
$\alpha = 7; \beta = 5$
$\alpha = 7; \beta = 11$
$\alpha = 7; \beta = 8$
$\alpha = 8; \beta = 7$
10
5
7
11
12
8
9
8
5
12
11
12
9
8
7
10
MAX
MIN
MAX
MIN
Terminal Nodes

Constraing Satisfaction Problems - CSP

Sudoku

- A Sudoku consist of Nine 3x3 boxes of cells.
- Each cell must contain a number from 1 to 9.

Variables: V = E_{11}, E_{12}, ..., E_{19}, E_{21}, E_{22}, ..., E_{29}, ..., E_{91}, E_{92}, ..., E_{99}

Domain: 1, 2, 3, 4, 5, 6, 7, 8, 9

Constraints:

- Alldiff(E_{11}, E_{12}, E_{13}, E_{14}, E_{15}, E_{16}, E_{17}, E_{18}, E_{19})
- ...
- Alldiff(E_{11}, E_{21}, E_{31}, E_{41}, E_{51}, E_{61}, E_{71}, E_{81}, E_{91})
- ...
- Alldiff(E_{11}, E_{12}, E_{13}, E_{14}, E_{15}, E_{16}, E_{17}, E_{18}, E_{19})
- ...

Sudoku - Example

A Sudoku and its solution are obtainted by applying the Constraints on the Variables.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Thank You