RE56 Labs
Specification document model – DELIVER 10 pages AT MAXIMUM

**STUDENTS**

DERRINGER ALINE
INACIO DAVID
GHNASSIA CLÉMENT

**WORK SUBJECT**

# HO: HANDOVER AND SELECTION/RESELECTION IN GSM/UMTS

**DOCUMENT VERSION**

04/23/2014: First version of the document

**TEACHER COMMENTS**

Date:
Comments:

---

## 1. PURPOSE OF THE SYSTEM FUNCTION

The handover and selection/reselection functionalities are needed to ensure mobility of users. Indeed, when a user move and change location, the network must be able to update the selected cell so the user does not lost the network and stay with a good connection. Handover and selection/reselection cover around the same needs but are used in different circumstances.

Selection is needed when the mobile has to select its first cell. It means that it's not currently connected to any network (the user switched it on or the network was lost because a lack of coverage). Reselection is used when the mobile is already connected and either idle or in packet switching (data transmission).

Handover is needed when the mobile user is currently used in circuit switching (classic voice call). Handover can be inter-system or intra-system. It is qualified by soft or hard depending on the conditions.

## 2. INPUT DATA AND PARAMETERS

- *is_calling a boolean indicating if the user is trying to make a call (in circuit switcing)*

- *hysteresis_umts : a number to define an hysteresis between two umts signal strength. Avoid flapping scenarii.*
- *hysteresis_gsm : a number to define an hysteresis between two gsm signal strengths. Avoid flapping scenarii*
- *ntimes : a number indicating how many times the check has to be done before performing the operation (handover or reselection). Avoid flapping scenarii.*
- *update_strength(cell(s)) : Update the strength of the cells given in parameters at the mobile location.*
- *measure_landscape(type) is a function that returns the cells of desired type(UMTS or GSM) which have been detected at mobile location.*

# 3. DESCRIPTION OF FORMULA AND ALGORITHM

## Free space propagation Friis formula

Calculate the power received at the mobile from an antenna depending on distance and frequency used

$Pr = Pe + Ge + Gr – (32{,}44 + 20 \log(F) + 20\log(d))$

*Pr = Power received at the mobile (in dBm)*
*Pe = Power emitted by the base station (in dBm)*
*Ge = Gain at base station in (dBi)*
*Gr = Gain at mobile station in (dBi)*
*F = Frequency of the used wave (in MHz)*
*d = Distance between the mobile and the base station (in km)*

## Algorithm

```
function update_strength
input
      List<Cell> cell_set
      mobile
output
      List<Cell> cell_set
body
      distance ← 0
      for cell in cell_set do
            distance ← sqrt(pow(abs(mobile.x - cell.x),2) + pow(abs(mobile.y -
cell.y), 2))
            cell.strength ← cell.signal + cell.gain + mobile.gain - (32,44 +
20log(cell.frequency) + 20log(distance))
      end do
end body
end function
```

```
function selectionUMTS
      input
            List<Cell> candidate_set
      output
            Cell selected
      body
            for cell in candidate_set do
                  if cell.strength >= cell.minimum_strength && ! cell.is_barred
then
```

```
                        selected ← cell
                        break
                  end if
            end do
      end body
end function
```

```
function selectionGSM
      input
            List<Cell> candidate_set
      output
            Cell selected
      body
            candidate_set.sort('desc')
            for cell in candidate_set do
                  if cell.strength >= cell.minimum_strength && ! cell.is_barred
then
                        selected ← cell
                  end if
            end do
      end body
end function
```

```
function reselectionUMTS
      input
            Cell selected
            List<Cell> candidate_set
            hysteresis
            ntimes
      output
            Cell selected
      body
            candidate_set.sort('desc')
            for cell in candidate_set do
                  if ! cell.is_barred then
                        if selected.type equals 'GSM' then
                              if cell.strength > cell.minimum_strength then
                                    cell.ntimes ← cell.ntimes + 1
                                    if cell.ntimes equals ntimes then
                                          selected ← cell
                                    end if
                              else
                                    cell.ntimes ← 0
                              end if
                        else  // selected.type equals 'UMTS'
                              if cell.strength > cell.minimum_strength &&
cell.strength - selected.strength > hysteresis then
                                    cell.ntimes ← cell.ntimes + 1
                                    if cell.ntimes equals ntimes then
                                          selected ← cell
                                    end if
                              else
                                    cell.ntimes ← 0
                              end if
                        end if
                        break
                  end if
            end do
      end body
end function
```

```
//we assume that entering to this function means that there is no UMTS //cell
available
function reselectionGSM
      input
            Cell selected
            List<Cell> candidate_set
```

```
            hysteresis
      output
            Cell selected
      body
            candidate_set.sort('desc')
            for cell in candidate_set do
                  if ! cell.is_barred then
                        if selected.type equals 'GSM' then
                              if  cell.strength  >  cell.minimum_strength  &&
cell.strength - selected.strength > hysteresis then
                                    cell.ntimes ← cell.ntimes + 1
                                    if cell.ntimes equals ntimes then
                                          selected ← cell
                                    end if
                              else
                                    cell.ntimes ← 0
                              end if
                        else  // selected.type equals 'UMTS'
                              if cell.strength > cell.minimum_strength
                                    cell.ntimes ← cell.ntimes + 1
                                    if cell.ntimes equals cell.ntimes then
                                          selected ← cell
                                    end if
                              else
                                    cell.ntimes ← 0
                              end if
                        end if
                  end if
            end do
      end body
end function
```

```
function handoverUMTS
input
      List<Cell> active_set
      List<Cell> neighbor_set
      hysteresis
      ntimes
output
      List<Cell> active_set
      List<Cell> neighbor_set
body
      for cell in active_set do
            maxActive ← max(active_set)
            if cell.strength < cell.minimum_strength then
                  active_set.remove(cell)
                  neighbor_cell.add(cell)
            else if maxActive.strength - cell.strength > hysteresis then
                  cell.ntimes ← cell.ntimes + 1
                  if cell.ntimes equals ntimes then
                        active_set.remove(cell)
                        neighbor_cell.add(cell)
                  end if
            else
                  cell.ntimes ← 0
            end if
      end do

      neighbor_set.sort('desc')
      for cell in neighbor_set do
            if active_set.length < active_maxlength then
                  if active_set.length equals 0 then
                        maxActive ← 0
                  else
                        maxActive ← max(active_set)
                  end if

                  if  cell.strength  >  cell.minimum_strength  and  cell.strength  -
maxActive.strength > hysteresis then
```

```
                        cell.ntimes ← cell.ntimes + 1
                        if cell.ntimes equals ntimes then
                              neighbor_cell.remove(cell)
                              active_set.add(cell)
                        end if
                  else
                        cell.ntimes ← 0
                  end if
            else
                  break
            end if
      end do

      isSwitched ← true
      while isSwitched do
            minActive ← min(active_set)
            maxNeighbor ← max(neighbor_set)
            if maxNeighbor.strength - minActive.strength > hysteresis then
                  maxNeighbor.ntimes ← maxNeighbor.ntimes + 1
                  if maxNeighbor.ntimes equals ntimes then
                        active_set.replace(minActive, maxNeighbor)
                        neighbor_set.replace(maxNeighbor, minActive)
                  end if
            else
                  maxNeighbor.ntimes ← 0
                  isSwitched ← false
            end if
      end do
end body
end function
```

```
function handoverGSM
input
      Cell primary
      List<Cell> neighbor_set
      hysteresis
      ntimes
      isHandoverNeeded
output
      Cell primary
      List<Cell> neighbor_set
body
      neighbor_set.sort('desc')
      for cell in neighbor_set do
            if not cell.is_barred && cell.strength > cell.minimum_strength
                  if primary is null then
                        cell.ntimes ← cell.ntimes + 1
                  else if cell.strength - primary.strength > hysteresis then
                        cell.ntimes ← cell.ntimes + 1
                  else
                        cell.ntimes ← 0
                  end if

                  if cell.ntimes greater than ntimes and isHandoverNeeded then
                        primary ← cell
                  end if
            else
                  cell.ntimes ← 0
            end if
      end do
end body
end function
```

```
function main
input
      boolean is_calling
      int hysteresis_umts
```

```
      int hysteresis_gsm
      int ntimes
body
      Cell selected
      Cell primary
      List<Cell> active_set
      neighbor_umts_set
      neighbor_gsm_set
      boolean is_update_calling ← false

      while 1 do
            update_strength(selected)
            update_strength(primary)
            update_strength(active_set)
            update_strength(neighbor_umts_set)
            update_strength(neighbor_gsm_set)

            if selected is null then //selection

                  candidate_set ← measure_landscape('UMTS')
                  selected ← selectionUMTS(candidate_set)
                  if selected is null then
                        candidate_set ← measure_landscape('GSM')
                        selected ← selectionGSM(candidate_set)
                  end if

                  if selected is not null then
                        neighbors_umts_set ← selected.getNeighborsUMTS()
                        neighbors_gsm_set ← selected.getNeighborsGSM()
                  end if
            else
                  if is_calling then
                        if not is_update_calling then
                              if selected.type equals 'UMTS'
                                    active_set.add(selected)
                              else
                                    primary ← selected
                              end if
                              is_update_calling ← true
                        end if

                        if primary equals null and active_set.length equals 0 then
                              is_calling ← false
                        else
                              handoverUMTS(active_set, neighbor_umts_set,
hysteresis_umts, ntimes)

                              handoverGSM(primary, neighbor_gsm_set,
hysteresis_gsm, ntimes, active_set.length equals 0)

                              if active_set.length > 0 and selected is not
max(active_set)
                                    primary ← null
                                    tmp_selected ← max(active_set)

                              else if selected is not primary then
                                    tmp_selected ← primary
                              end if

                              if tmp_selected not equals selected then
                                    selected ← tmp_selected
                                    neighbors_umts_set                    ←
selected.getNeighborsUMTS().minus(active_set)
                                    neighbors_gsm_set ← selected.getNeighborsGSM()
                              end if
                        end if

                  else
                        if is_update_calling then
```

```
                              is_update_calling ← false
                              if not active_set.is_empty() then
                                    active_set.empty()
                              else
                                    primary ← null
                              end if
                        end if
                        Cell tmp_selected
                        tmp_selected ← reselectionUMTS(selected,
neighbor_umts_set, hysteresis_umts, ntimes)
                        if tmp_selected.type is null then
                              tmp_selected ← reselectionGSM(selected,
neighbor_gsm_set, hysteresis_gsm, ntimes)
                        end if
                        if tmp_selected not equals selected then
                              selected ← tmp_selected
                              neighbors_umts_set ← selected.getNeighborsUMTS()
                              neighbors_gsm_set ← selected.getNeighborsGSM()
                        end if
                  end if
            end if
            sleep(500)  //wait for 0.5 sec
      end do
end body
end function
```

## 4.OUTPUT DATA

The main function should simulate the operations that the mobile or network must take depending on the situation. The function performs simulation of selection/reselection on GSM and UMTS, soft handover (UMTS to UMTS) and hard handover (UMTS to GSM, GSM to UMTS, and GSM to GSM).

The selected cell is only used when the user is not calling. When the user gives a call, the content of the selected cell is inserted either in the active_set list (if the selected cell is UMTS) or in the primary variable (if the selected cell is GSM).

## 5. VALIDATION ON ONE EXAMPLE

For the example we are going to take following parameters:

*is_calling* ← *false*
*hysteresis_umts* ← *4*
*hysteresis_gsm* ← *2*
*ntimes* ← *3*

At the beginning all the sets are empty and the variables primary and selected are null, so we enter in the first condition in the main function.

We now invoke the measure_landscape with UMTS parameters. This function simply return a set of detected UMTS cells associated with their signal strengthes depending on the distance between each cells and the mobile phone. Let's suppose that the function return the following measures:

*candidate_set* ← *{Cell1: 5, Cell2: 11, Cell3: 15}*

Now, in order to select an appropriate cell, we invoke selectionUMTS function passing the candidate_set. If we suppose that the minimum_strength needed to select the cell is 10 and no one is barred, the function will return the first cell with a strength greater or equal than the minimum strength required. In this case, the cell returned is Cell2, with type UMTS. From this cell (cell(2)), we retrieve the neighbors (in UMTS and GSM). Let's suppose we have:

*selected ← (Cell2: 11)*
*neighbor_umts_set ← {Cell3, Cell 4, Cell6}*
*neighbor_gsm_set ← {Cell11, Cell12}*

In the next iteration in the loop. We implement measurement of the neighbor cells as well as the selected cell. We assume that we get the following measures:

*selected ← (Cell2: 11)*
*neighbor_umts_set ← {Cell3: 15, Cell 4:18, Cell6:8}*
*neighbor_gsm_set ← {Cell11:18, Cell12:16}*

Since we have a selected cell, we enter in the else statement. We are going to invoke the function reselectionUMTS(). For a reselection is being performed, the cell must be stable, which means that it has to match the conditions during three loops. We suppose that the values don't move during three loops, so the cell Cell4 will be the result of reselection. Indeed, the strength of Cell4 is greater or equal than Cell1 added with hysteresis for UMTS. Because we changed cell, the neighbor sets changed as well. We get from the reselected cell the following values and their associated strength (after we proceeded to measurements):

*selected ← (Cell4: 18)*
*neighbor_umts_set ← {Cell2: 11, Cell3:15, Cell6:8}*
*neighbor_gsm_set ← {Cell11:15, Cell16:8}*

Let's imagine that we want to give a call. First of all, if the cell is a UMTS cell, it's added in the active_set. Let's suppose we have now the following values:

*selected ← (Cell4: 18)*
*active_set ← {Cell4: 18}*
*neighbor_umts_set ← {Cell2: 11, Cell3:22, Cell6:8}*

Notice: Meanwhile, the neighbor_gsm_set is updated (the ntime is incremented for possible candidate), in case of interruption with UMTS connection.

If the results measurements are not changing for three loops. Now we retrieve the neighbors from the cell in the active set which has the maximum strength. The results will be eventually:

*active_set ← {Cell4: 18, Cell3:22}*
*neighbor_umts_set ← {Cell5: 11, Cell6:8}*
*neighbor_gsm_set ← {Cell12:14, Cell13:18}*

And if the measurements are not changing for more three loops, the Cell4 will be removed from the active set and added in the neighbor set; the selected cell is now referencing the Cell3 (because it is in the max strength in the active_set.

*selected ← (Cell3: 22)*
*active_set ← {Cell3:22}*
*neighbor_umts_set ← {Cell4: 18, Cell5:11, Cell6:8}*

Now, if the mobile is moving and all the UMTS's cells have a strength lower than the minimum required (10); a hard handover towards GSM cell is needed. We assume that the Cell13 has already a "ntimes" greater than the "ntimes" required to be qualified as stable (don't forget that the handoverGSM treatment has been performed since we are in a call).

*selected ← (Cell3: 9)*
*active_set ← {Cell3:9}*
*neighbor_umts_set ← {Cell2: 8, Cell4: 7, Cell6:9}*
*neighbor_gsm_set ← {Cell12:14, Cell13:18}*

After the iteration in the loop, we performed hard handover and retrieve neighbors of the new primary cell.

*selected ← (Cell13: 18)*
*primary ← { Cell13: 18}*
*active_set ← {}*
*neighbor_umts_set ← {Cell9: 9}*
*neighbor_gsm_set ← {Cell12:14, Cell19:8}*

The communication is stopped, we juste remove the primary and optionnally empty the active_set (which in this case was already empty since the communication was under GSM).

*selected ← (Cell13: 18)*
*primary ← {}*
*active_set ← {}*
*neighbor_umts_set ← {Cell9: 9}*
*neighbor_gsm_set ← {Cell12:14, Cell19:8}*

Because the mobile is now idle, the algorithm will return to selection/reselection mode.

# 6. BIBLIOGRAPHY USED


- UMTS, services, architectures et WCDMA, *Javier Sanchez and Mamadou Thioune*, Hermes
- Réseaux mobiles, *Sami Tabbane*, Hermes
- Simulating soft handover and power control for enhanced umts, *V. Vassiliou, J. Antoniou, A. Pitsillides, and G. Hadjipollas,* Department of Computer Science in University of Cyprus