R1-080719: Hopping Patterns for UL RS

Agenda Item: 6.1.2

Group Sequence Hopping (when enabled)

- **RAN1#51bis (Sevilla) Agreements**
  - **Two layered Hopping/Shifting of Sequence Groups (R1-080241)**
    - Pattern generated by combining (17) root hopping pattern and (30) sequence shift
    - Common root hopping pattern among uplink RSs (DM RS, SRS, PUCCH)
    - Implicit cell-ID based group-assignment (or group hopping index) for the PUCCH/SRS
    - PUSCH DM RS group-assignment (or group hopping index) signaled (5-bit sequence shift relative to that for PUCCH/SRS)
- **The 30 sequence shifts are {0, 1, 2, … 29} and independent of the slot index**
  - **fss(n_s) = k, k is from set {0, 1, 2, … 29}**
- **Need to define 16 root group hopping patterns (pattern 1 – 16)**
  - **Root Pattern 0 is the all zero group index corresponding to no hopping**
- **The root hopping pattern can be generated from the output of a PN generator (scrambling sequence generator)**
  - **Root hopping pattern = (5-bits O/P) modulo 30**
- **Initialization of the 31-bit scrambling generator (MSB bit = 1)**
  - **4 (root pattern index) + 5 (slot index) + 2 (SFN LSBs)**
  - **Period of the root hopping pattern = 40ms = 80 slots**
- **Re-initialization period = slot**

Sequence Hopping Pattern with the Group

- **When group hopping is disabled (i.e. sequence planning)**
  - **Same PUSCH/SRS/PUCCH group assignment (5-bit signaled)**
  - **Sequence hopping within the RS group (>5RBs) can also be disabled (Jeju)**
    - One of the two base sequences v=0 or v=1 can be selected within the group
    - Allow for larger re-use distances
  - **When sequence hopping within the RS group is enabled**
    - Hopping pattern generated from the output of scrambling sequence generator
      - base sequence number = bit value, bit 0 = v0, bit 1=v1
    - Initialization of the 31-bit scrambling generator (MSB bit = 1)
      - 6/7 (Cell ID MSBs) + 5 (group assignment) + 5 (slot index) + 2 (SFN LSBs)
      - Function of partial cell-ID enables coordinated cell cluster with same hopping patterns for e.g. cells of a eNode-B
    - Period of the sequence hopping pattern = 40ms = 80 slots
    - Re-initialization period = slot
- **When group hopping is enabled**
  - **Can use same sequence hopping pattern (>5RBs) as above to provide possible additional randomization**
    - No increase in UE complexity as UE has to anyway use a different base sequence due to group hopping

Cyclic Shift Hopping for PUSCH DM RS

- **Cell may be assigned a subset of the available shifts with different shifts assigned to different cells of a Node-B (Kobe, RAN1#49)**
- **3 bits in the UL grant indicate up to 8 cyclic shifts per sequence**
- **Cyclic Shift (CS) hopping within the subset of shifts assigned to cell**
- **CS hopping offset pattern generated from the output of scrambling sequence generator**
  - **CS hopping offset pattern = 3-bits O/P (CS Index)**
  - **CS hopping pattern = (Assigned CS + CS offset pattern) modulo 8**
- **Initialization of the 31-bit scrambling generator (MSB bit = 1)**
  - **6/7 (Cell ID MSBs) + 5 (group assignment) + 5 (slot index) + 2 (SFN LSBs)**
    - Same CS hopping offset pattern for all UEs within a cell
    - Function of partial cell-ID enables for planning/coordination of hopping patterns
    - Coordination of shift values among the cells of a Node-B or adjacent Node-Bs using the same base sequence/group
- **Period of the sequence hopping pattern = 40ms = 80 slots**
- **Re-initialization period = slot**

Cyclic Shift Hopping for PUCCH (RS and data LBs)

- Cell-specific cyclic shift hopping per symbol
- Implicitly assigned initial cyclic shift for RS and control LBs for first slot
- Cyclic shift (CS) hopping within the subset of shifts allocable to the same orthogonal code
  - Possible shifts for a orthogonal code depends on the delta_shift, and delta_offset
  - Ncs_oc = number of cyclic shifts for a orthogonal code (determines number of A/N channels)
- Same CS hopping offset pattern for all (e.g. 6) cyclic shifts of a orthogonal code
  - CS hopping pattern = (Assigned CS + CS offset pattern) modulo Ncs_oc
- CS hopping offset pattern generated from the output of scrambling sequence generator
  - CS index hopping offset pattern = (4-bits O/P) modulo Ncs_oc
- Initialization of the 31-bit scrambling generator (MSB bit = 1)
  - 9 (Cell ID) + 4 (SC-FDMA symbol index) + 4 (sub-frame index) + 2 (SFN LSBs)
    - Same CS hopping offset pattern for both slots of a sub-frame
    - Same CS hopping offset pattern for different orthogonal codes (and without OC – e.g. CQI)
- Re-initialization period = SC-FDMA symbol
- Period of the sequence hopping pattern = 40ms = 80 slots
- Cyclic shift hopping between the two slots of a sub-frame
  - Remap implicitly assigned initial CS but same hopping offset pattern as first slot
  - Initial CS in second slot function of (initial CS in first slot)

Orthogonal Cover Hopping for PUCCH (RS and data LBs)

- **Implicitly assigned initial orthogonal cover for RS and control LBs for first slot**
- **Cell-specific Orthogonal code index hopping between slots**
- **OC hopping offset pattern generated from the output of scrambling sequence generator**
  - **OC index hopping offset pattern = (2-bits O/P) modulo Noc**
  - **Noc = number of orthogonal codes**
  - **OC hopping pattern = (Assigned OC index + OC offset pattern) modulo Noc**
- **Initialization of the 31-bit scrambling generator (MSB bit = 1)**
  - **9 (Cell ID) + 4 (sub-frame index) + 2 (SFN LSBs)**
    - Same OC hopping offset pattern for different orthogonal codes
- **Re-initialization period = Sub-frame**
- **Period of the sequence hopping pattern = 40ms = 80 slots**

Conclusions

- All hopping patterns (Group Root Hopping, Sequence Hopping, CS Hopping, OC Hopping) generated from scrambling sequence generator with different initialization bits and re-initialization periods
- PUSCH Sequence hopping patterns within the RS group should allow for same hopping patterns for e.g. cells of a eNode-B
- PUSCH cyclic shift hopping patterns should enable coordination of shift values among the cells of a Node-B or adjacent Node-Bs using the same base sequence/group
- For sequence planning and no sequence hopping within the RS group, it should be possible to select one of the two base sequences v=0 or v=1 to allow for larger re-use distances