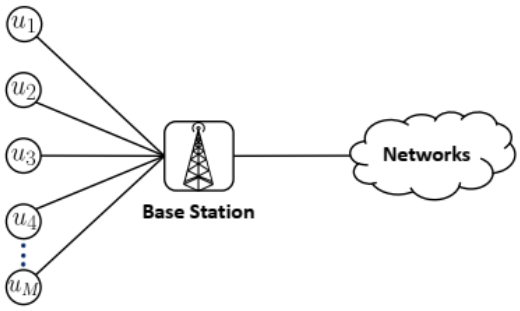
# BAB 3
# METODE PENELITIAN

## 3.1 Alat yang Digunakan

Penelitian ini menggunakan *software* dan *hardware*. *Software* yang digunakan adalah MATLAB R2018b dalam melakukan simulasinya. Sedangkan *hardware* yang digunakan adalah *Laptop* dengan spesifikasi *Intel Core* i3-10110U CPU 2.10 GHz dengan kapasitas 7,84 GB.

## 3.2 Model istem

Pada masa mendatang jaringan telekomunikasi *wireless* diprediksi akan memiliki tingkat lalu lintas pertukaran data yang padat seperti yang ditunjukkan pada Gambar 3.1 terutama *link* antara *user* dengan *base station* (BS). Sejumlah *user* dari $u_1$ hingga $u_M$ transmit ke BS untuk selanjutnya masuk ke jaringan internet. Jumlah transmisi dari *user* ke BS dibentuk dari *degree distribution*.



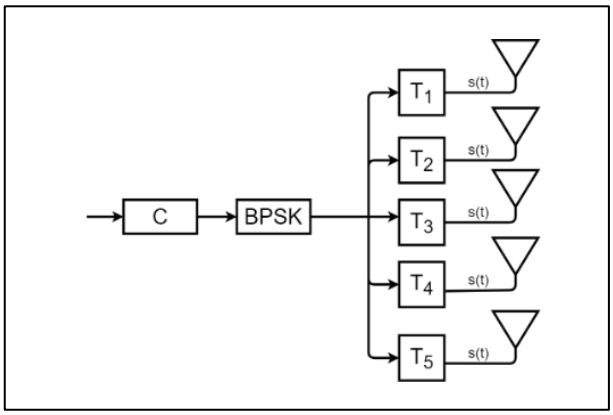

**Gambar 3.1 Struktur jaringan Super Padat dengan Jumlah *User* yang Sangat Besar[3]**

Gambar 3.1 menunjukkan skema komunikasi antara beberapa *user* dengan *base station*. Terlihat pada gambar tesebut bahwa setiap *user* mengirimkan paket informasi masing-masing ke *base station* sebelum dilanjutkan ke *network*. Jumlah transmisi dari *user base station* akan menghasilkan *degree distribution* yang dilihat dari *user* dan *time slot*. Dengan *degree distribution* yang optimal, jaringan yang diusulkan diharapkan dapat mencapai kapasitas jaringan yang terkait.

### 3.2.1 *Transmitter*

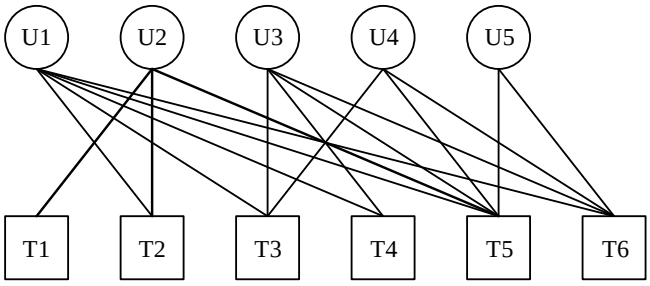
Jumlah transmisi dari *user* ke *Base Station* (BS) membentuk sebuah transmisi *degree distribution* dari sudut pandang *user* dan *time-slot*. Proses SIC

(*Successive Interference Cancellation*) di dalam BS dan sistem prioritas ada pada *receiver* (BS). Sistem IoT pada penelitian ini membagi *user* dalam satu kategori. *Transmitter* menggunakan *single carrier transmission* untuk menjaga sistem tetap sederhana karena *equalizer* tidak dibutuhkan. *Code* paket dimodulasi menggunakan modulasi BPSK. Pada penelitian ini, *repetition codes* didesain untuk komunikasi yang besar antara *user* dan BS. *Repetition codes* digunakan pada data dengan kecepatan rendah misalnya utilitas jaringan sensor nirkabel di dalam mesin.




**Gambar 3.2 Contoh *Transmitter* untuk *Degree* $d = 5$**

Sistem model pada penelitian ini diilustrasikan pada Gambar 3.3 yang sejumlah *user* mengakses *time-slot* secara bebas dan acak (*random*). Lingkaran mewakili *user node* (UN) yang memiliki *degree* $\ell$ sedangkan kotak mewakili *slot node* (SN) yang memiliki *degree* $d$. *Erasure probability* yang keluar dari *user* direpresentasikan dengan $q$ dan *erasure probability* yang keluar dari *time-slot* direpresentasikan dengan $p$.




**Gambar 3.3 *Bipartite graph* CRA untuk menampilkan EXIT *Chart*, PLR dan *Throughput***

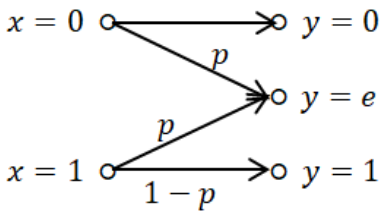
UN ditunjukkan dengan $U_h = \{U_1, U_2, \ldots\ldots, U_5\}$ dan total *time-slot* $T = \{t_1, t_2, \ldots\ldots, t_6\}$. Target dari sistem skema CRA adalah untuk mendapatkan *degree*

*distribution* yang optimal. Pada sistem ini, *user* akan transmit ke *time-slot* yang sama, namun tidak mengetahui identitas serta paket dari *user* mana yang transmit.

*Bipartite graph* seperti yang ditunjukkan pada Gambar 3.3 juga digunakan untuk melihat grafik *Throughput*, PLR dan EXIT *Chart*. Nilai PLR didapatkan dengan membandingkan jumlah informasi yang diterima oleh *receiver* setelah melewati kanal dengan informasi awal yang dikirimkan. Nilai *Throughput* dihasilkan dari rasio jumlah paket benar yang diterima tanpa adanya *error* dengan jumlah paket yang dikirimkan termasuk paket yang salah.

### 3.2.2 Kanal yang Digunakan

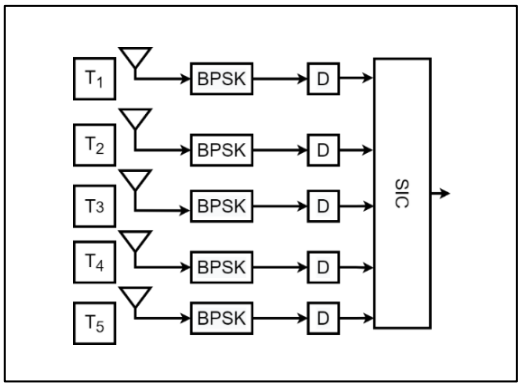
Penelitian ini menggunakan kanal *Binary Erasure Channel* (BEC) untuk menghasilkan EXIT *Chart*. Dalam satu *link* komunikasi, terdapat kemungkinan bahwa data yang diterima oleh *receiver* mengandung data yang salah (*error*)[35], Gambar 3.4 menunjukkan BEC, dengan *erasure probability P*, yaitu probabilitas data *x* hilang dalam transmisi. BEC memiliki nilai *P* yang biasanya berkisar pada $0 \leq p \leq 1/2$, dengan model masukan yaitu $x_1$ dan $x_2$ dan tiga keluaran yaitu $y_1$, $y_2$, dan $y_3$. Jika dikirimkan paket "0" probabilitas yang diterima adalah "e", hal yang sama berlaku untuk bit "1"[1].




**Gambar 3.4 BEC dengan *Erasure probability p*[1]**

### 3.2.3 *Receiver*

Pada *receiver* terdapat beberapa proses yang diperlukan untuk mendapatkan pesan. Setelah melewati kanal, sinyal didemodulasi menggunakan demodulasi BPSK serta deteksi identitas paket yang menunjukkan paket *time-slot* sudah ditransmisikan. Simbol dikodekan dengan SIC berdasarkan pada grafik faktor. CRA mengasumsikan sinkronisasi dan deteksi *header* yang sempurna untuk menunjukkan SIC dalam skema *decoding* pada *receiver*. SIC dapat dengan mudah ditunjukkan dengan mengurangi pesan yang sudah ditentukan dari pesan yang diterima sehingga gangguan dapat dihilangkan, yang berarti *degree* SN berkurang.

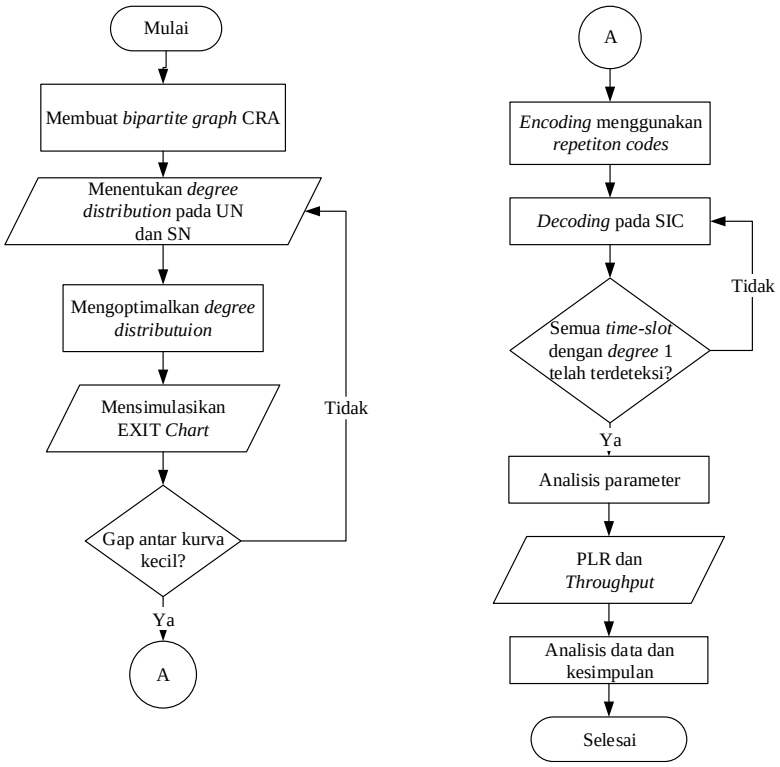



**Gambar 3.5 Contoh *Receiver* untuk *Degree* $d = 5$**

SIC merupakan salah satu metode manajemen interferensi. Metode ini menggunakan NOMA sebagai deteksi *multi-user* pada *receiver*. Dua atau lebih sinyal dari *user* yag berbeda yang dialokasikan pada *subcarrier* yang sama pada saat transmisi akan saling menginterferensi satu sama lain. Dengan menerapkan SIC pada masing-masing *receiver*, sinyal dari *user* lain yang menginterferensi diseleksi sehingga didapat sinyal yang diharapkan.

## 3.3 Alur Penelitian

Penelitian ini dapat dilakukan dengan mengacu pada Gambar 3.6:




**Gambar 3.6 Diagram alur penelitian**

Diagram alur penelitian pada Gambar 3.6 dapat diuraikan sebagai berikut:

1. Membuat *bipartite graph* CRA

   Tahap pertama yang dilakukan yaitu membuat *bipartite graph* yang akan digunakan sebagai acuan dalam penelitian. *Bipartite graph* terdiri dari *user* dan *time-slot*.

2. Menentukan *degree distribution* pada UN dan SN:

   Dari *bipartite graph* yang telah dibuat sebelumnya, dapat dihitung *degree distribution* pada UN dan SN. Untuk UN, terdapat dua tahap yaitu menghitung *node perspertive degree distribution* kemudian menghitung *edge perspective degree distribution*. Sedangkan untuk SN dilakukan aproksimasi probabilitas pada SN yang memiliki *degree d*.

3. Mengoptimalkan *degree distribution*

   *Degree distribution* yang telah didapatkan dioptimalisasi sebelum mensimulasikannya pada *software* Matlab.

4. Mensimulasikan EXIT *Chart*

   Hasil optimalisasi *degree distribution* disimulasikan pada *software* Matlab untuk mendapatkan grafik EXIT *Chart*.

5. Gap antar kurva

   Meganalisis apakah *gap* antar kurva yang didapatkan dari hasil simulasi EXIT *Chart* menunjukkan jarak yang kecil atau tidak. Apabila *gap*-nya kecil maka akan berlanjut ke proses yang berikutnya. Namun, apabila *gap*-nya besar maka harus menentukan *degree distribution* pada UN dan SN kembali.

6. *Encoding* menggunakan *repetition codes*

   Pada bagian ini terjadi proses *encoding* dengan menggunakan *repetition codes*.

7. *Decoding* pada SIC

   Pada proses ini, *user* yang terhubung dengan *time-slot* yang mempunyai *degree* 1 akan ter-*decoding* yang artinya semua paket yang dikirimkan oleh *user* tersebut telah mengirimkan paket tanpa terjadi *error*.

8. Semua *time-slot* dengan *degree* 1 telah terdeteksi?

   Jika semaua *time-slot* yang mempunyai *degree* 1 telah terdeteksi semua maka akan berlanjut ke proses yang berikutnya. Namun, apabila masih ada *time-slot* yang belum terdeteksi maka harus kembali pada proses *decoding* SIC.

9. Analisis parameter

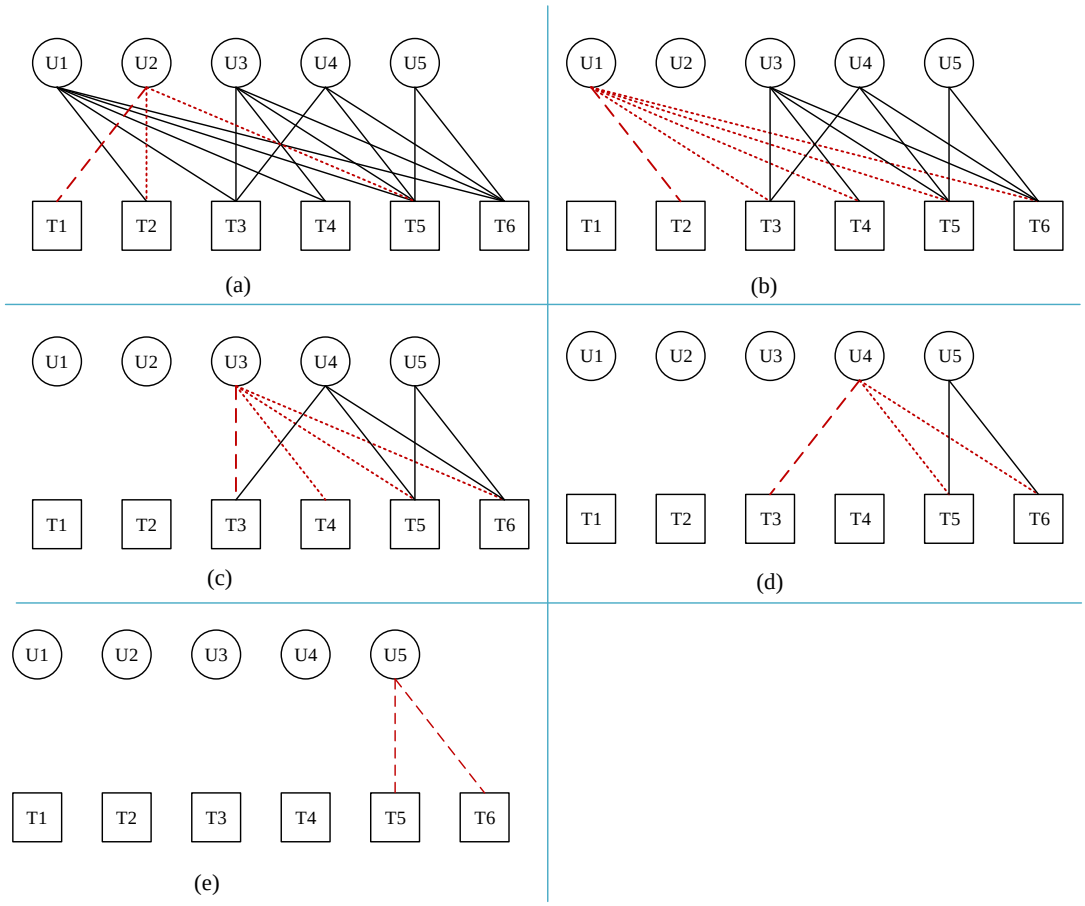
   Menganalisis parameter yang digunakan dalam penelitian, yaitu PLR dan *Throughput* yang dihasilkan dari simulasi.

10. Analisis data dan kesimpulan

    Menganalisis hasil dari simulasi yaitu EXIT *Chart*, PLR, dan *Throughput*, serta membuat kesimpulan dari analisis tersebut.

## 3.4 Proses Decoding SIC

Gambar 3.7 adalah skema algoritma yang digunakan dalam tahapan *Decoding* dari *Repetition codes*. Gambar dari skema tersebut adalah sebagai berikut:




**Gambar 3.7 Algoritma *decoding* SIC pada *receiver***

Langkah yang pertama yaitu mencari *user* $u_i$ yang mempunyai *degree* satu. Hilangkan sinyal yang diterima oleh semua *time-slot* $t_t$ yang terhubung dengan *user* $u_i$ sehingga semua transmisi sinyalnya berhasil terdeteksi. Pada Gambar 3.7 (a)

menunjukkan *user* U2 yang mempunyai *degree* 1 yang terhubung dengan *time-slot* T1. Hal ini menunjukkan semua informasi yang dikirimkan oleh *user* U2 telah diterima dengan benar tanpa adanya *error*. Proses tersebut akan diulangi hingga tidak ada lagi *degree* 1. Apabila tidak ada lagi *degree* 1 pada semua *time-slot* maka proses *decoding* akan berhenti seperti pada Gambar 3.7 (e). Hal ini menunjukkan bahwa semua *user* telah berhasil dideteksi.