Date:2022-01-01

Continue with my interim report. Finish the pre-matching algorithm section,

4.2.3 Pre-matching algorithm

To separate all the SWIPT-Supported D2D links and Non-EH D2D links from the D2D link set D, a pre-matching algorithm will be proposed to achieve that.

As mentioned in the section 5.2, to separate the SWIPT-supported D2D link, the EH sensitivity needs to be taken into consideration. In this project, the standard for checking if the current D2D i can perform SWIPT is its minimum power splitting ratio $\theta_{i,min}$ and maximum Throughput $T_{i,max}^{D}$. $\theta_{i,min}$ can be calculated as:

$$\theta_{i,min} = \frac{P_{thresold}^{1}}{P_{max}h_i^D + P_k^c h_{k,i} + N_0} \qquad (7)$$

According to equation (1), it can be transformed to:

$$T_i^D = \frac{P_i^D h_i^D}{P_k^c h_{k,i} + N_0 + \frac{N_1}{1-\theta_i}} \qquad (8)$$

Then the maximum Throughput for each D2D i can be easily achieved when θ_i reaches minimum:

$$T_{i,max}^D = \frac{P_i^D h_i^D}{P_k^c h_{k,i} + N_0 + \frac{N_1}{1-\theta_{i,min}}} \qquad (9)$$

In this project, for each SWIPT-Supported D2D link i, its minimum power splitting ratio should not be greater than 1 and its maximum throughput should meet the D2D minimum throughput requirement.

As shown in Algorithm 1, the generated D2D links set D, CUE set C, transmission power of CUE P_k^c, the minimum power segment $P_{thresold}^{1}$, maximum transmission P_{max}, the minimum Throughput of a D2D link T_{min}^D will be taken as input, and it should generate a partner selection PS, a SWIPT-supported D2D link group $EnaD$ and a Non-EH group $InfD$.

To start with, first initialize two empty sets for $EhaD$ and $InfD$ respectively. Each D2D link i will be paired with a CUE k from a sub-partner selection PS_i^D which will be initialized as the CUE set C for each loop of D2D link i. For each loop of CUE, the minimum power splitting ratio $\theta_{i,min}$ and the maximum throughput $T_{i,max}^D$ of each D2D i can be obtained using equation (7) and (9) respectively. Then, as mentioned before, for each D2D link i paired with the current CUE k, if the minimum power splitting ratio $\theta_{i,min}$ of the current D2D i is greater than 1 and the maximum throughput $T_{i,max}^D$ is smaller than the minimum throughput T_{min}^D, that means it is impossible for the current CUE k to help the D2D link i perform SWIPT, then the current CUE k will be removed from the sub-partner selection set. Finally, at the end of loop of the CUE set, the current sub-partner selection will be

To start with, first initialize two empty sets for $EhaD$ and $InfD$ respectively. Each D2D link i will be paired with a CUE k from a sub-partner selection PS_i^D which will be initialized as the CUE set C for each loop of D2D link i. For each loop of CUE, the minimum power splitting ratio $\theta_{i,min}$ and the maximum throughput $T_{i,max}^D$ of each D2D i can be obtained using equation (7) and (9) respectively. Then, as mentioned before, for each D2D link i paired with the current CUE k, if the minimum power splitting ratio $\theta_{i,min}$ of the current D2D i is greater than 1 and the maximum throughput $T_{i,max}^D$ is smaller than the minimum throughput T_{min}^D, that means it is impossible for the current CUE k to help the D2D link i perform SWIPT, then the current CUE k will be removed from the sub-partner selection set. Finally, at the end of loop of the CUE set, the current sub-partner selection will be checked if it is empty, if so, it means that the current D2D link i cannot find any CUE k to help it perform SWIPT hence cannot activate EH, then it will be added to $InfD$. Otherwise, it means that there is at least one CUE k which can help it perform SWIPT and it will be added to the group $EnaD$. The sub-partner selection will be added

to the partner selection set PS. So, at the end of this pre-matching algorithm, each D2D link i will have a partner selection set where the CUEs can help it perform SWIPT.

It is noted that each CUE k can be selected in more than one partner selection of different SWIPT-Supported D2D links after using the pre-matching algorithm, and for each SWIPT-Supported D2D link i, it can have more than one CUE in its partner selection. To further increase the accuracy of the matching process between each SWIPT-Supported D2D link i and CUE k to maximize the sum EE of each SWIPT-Supported D2D link I, each D2D link i needs to find its best CUE partner based on its partner selection.

ALGORITHM 1: PRE-MATCHING ALGORITHM

Algorithm 1 Pre-Matching Algorithm		
Input	:	$D, C, P_k^C, P_{threshold}^1, P_{max}, T_{min}^D$
output	:	$PS, InfD, EnaD$
Initialize	:	$EnaD = \emptyset, InfD = \emptyset$
Step 1	:	for $i \in D$ do
Step 2	:	$PS_i^D = C$
Step 3	:	for $k \in C$ do
Step 4	:	obtain $\theta_{i,min}$ using (6), obtain $T_{i,max}^D$ using (8)
Step 5	:	if $\theta_{i,min} \geq 1$ or $T_{i,max}^D \leq T_{min}^D$ then
Step 6	:	Remove current k from PS_i^D
Step 7	:	end if
Step 8	:	end for
Step 9	:	if PS_i^D=$\emptyset$ then
Step 10	:	add i to $InfD$
Step 11	:	elseif $PS_i^D \neq \emptyset$ then
Step 12	:	add i to $EnaD$
Step 13	:	end if
Step 14	:	add PS_i^D to PS
Step 15	:	end for

Give the pseudo code for this algorithm based on my understanding.

Algorithm 1 Pre-Matching Algorithm

Input	:	$D, C, P_k^C, P_{thresold}^1, P_{max}, T_{min}^D$
output	:	$PS, InfD, EnaD$
Initialize	:	$EnaD = \emptyset, InfD = \emptyset$
Step 1	:	for $i \in D$ do
Step 2	:	$PS_i^D = C$
Step 3	:	for $k \in C$ do
Step 4	:	obtain $\theta_{i,min}$ using (6), obtain $T_{i,max}^D$ using (8)
Step 5	:	if $\theta_{i,min} \geq 1$ or $T_{i,max}^D \leq T_{min}^D$ then
Step 6	:	Remove current k from PS_i^D
Step 7	:	end if
Step 8	:	end for
Step 9	:	if $PS_i^D = \emptyset$ then
Step 10	:	add i to $InfD$
Step 11	:	elseif $PS_i^D \neq \emptyset$ then
Step 12	:	add i to $EnaD$
Step 13	:	end if
Step 14	:	add PS_i^D to PS
Step 15	:	end for

And the implemented code for pre-matching algorithm is given as:

```
function [SiD,InfD,EhaD,h_D2D,h_C_D2D,h_D2D_BS,h_CUE] =
Prematch(D,C,Pkc,Pth1,Pmax,TminD,distance_D2D)
syms rayleigh_ki rayleigh_i_D
syms lambda_min T_max
EhaD=[];
InfD=[];
count_delete=0;
%pass loss exponent
pass_loss=3;
%N0 N1
N0=1*10^(-13);
N1=1*10^(-13);

%Set hDi hki hkc hiB
h_D2D=[];
h_C_D2D=[];
h_D2D_BS=[];
h_CUE=[];

%initialize BS
BS=[0 0];

for i=1:size(D,1)
    SiD{i,1}=C;
    count_delete=0;
    hD=exprnd(1)/(distance_D2D^(pass_loss));
    D2D_TX=D{i,1};
    D2D_RX=D{i,2};
    dis_i_BS=point_to_line(D2D_TX,D2D_RX,BS);
    hDB=exprnd(1)/(dis_i_BS^(pass_loss));
    h_D2D(i,1)=hD;
```

```
    h_D2D_BS(i,1)=hDB;
    for k=1:size(C,1)
        CUE_point=C(k,:);
        v1=D{i,1};
        v2=D{i,2};
        dis_k_D2D=point_to_line(v1,v2,CUE_point);
        hki=exprnd(1)/(dis_k_D2D^(pass_loss));

        dis_k_BS=hypot(CUE_point(1),CUE_point(2));
        hkc=exprnd(1)/(dis_k_BS^(pass_loss));
        h_CUE(k,1)=hkc;
        lambda_min=(Pth1)/(Pmax*hD+Pkc*hki+N0);
        T_max=log2(1+(Pmax*hD)/(Pkc*hki+N0+(N1)/(1-lambda_min)));
        h_C_D2D(i,k)=hki;
        if  lambda_min>1 || T_max<=TminD
            temp=SiD{i,1};
            temp(k,:)=[0 0];
            SiD{i,1}=temp;
            count_delete=count_delete+1;
        end
    end

    if  count_delete==k
        InfD(end+1)=i;
    elseif count_delete<k
        EhaD(end+1)=i;
    end

end
end
```