THE 23rd ECMI MODELLING WEEK
EUROPEAN STUDENT WORKSHOP
ON MATHEMATICAL MODELLING
IN INDUSTRY AND COMMERCE

Wrocław University of Technology, Wrocław, Poland, August 23-30, 2009

Report of project group 10 on

How to finish a cycling race in shortest possible time?

The 23^{rd} ECMI Modelling Week
Wroclaw University of Technology

How to finish a cycling race in shortest possible time?

instructor:	Elisa Röhrig
students:	Christopher Jonsson, Shilan Mistry
	Przemyslaw Jurewicz, Nicole Noack
	Jarno Rantala, Reena Undla

Wroclaw 2009

Contents

1 Problem statement

The main aim of this project was to find the shortest time to finish a cycling race. It is assumed that the cyclist has a certain training level and that the track he has to finish is known. The main problem has been divided into two small ones:

1. Find the final time for a track if an arbitrary power distribution is given.
2. Find an optimal power distribution for a track that minimizes time.

The second problem also takes into account, that the cyclist gets exhausted while pedaling and recovers energy in some part of the track, for example when the cyclist is going downhill. Naturally the cyclist can not exceed a certain level of exhaustion. The data, which was used to compute the final time and model the power distribution of the cyclist is the track, shown in figure (1).

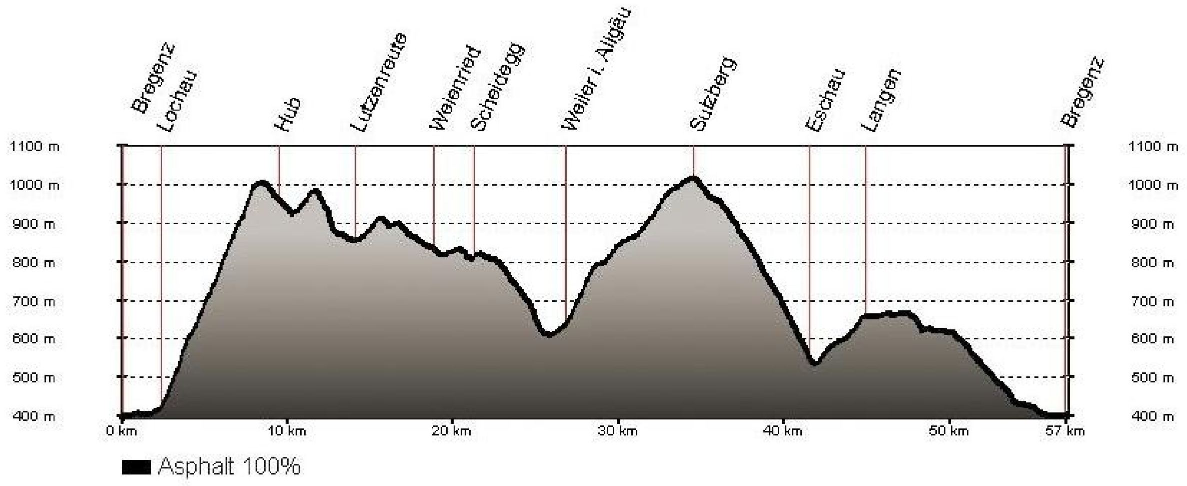


Figure 1: racing track

2 Basic equations

To ride a bicycle you have to overcome four basic forces: rolling resistance, air resistance, gravity and acceleration. The fundamental equation for power is force multiplied by velocity:

$$P = F_{total} \cdot v \tag{1}$$

where F_{total} consists of the sum of four forces:

$$F_{total} = F_{roll} + F_{wind} + F_{slope} + F_{accel}. \tag{2}$$

The equations for F_{roll}, F_{wind}, F_{slope} and F_{accel} are following:

$$\begin{aligned} F_{roll} &= c_r \cdot m \cdot g, & (3)\\ F_{wind} &= \frac{1}{2} r \cdot c_w \cdot A \cdot v^2, & (4)\\ F_{slope} &= s \cdot m \cdot g, & (5)\\ F_{accel} &= a \cdot m, & (6) \end{aligned}$$

where constants and coefficients are:

- c_r=0.003 (coefficient of rolling resistance, dimensionless)
- K=$c_w \rho A$, where ρ and $c_w A$ are
  ρ=1.2 kg/m^3 (density)
  $c_w A = 0.39 m^2$ (c_w is coefficient of wind resistance, dimensionless and A is frontal area, with unit m^2)
- m=80kg (total mass of the bicycle and cyclist)
- $g = 9.81 m/s^2$ (gravity)
- s: (slope, dimensionless)
- d: distance
- P: power

For the sake of simplicity it is assumed that F_{accel}=0. That's reasonable because the track is divided into small intervals, where the velocity won't vary much. So the acceleration force is very small compared to other forces, so it can be neglected. The formula for air resistance strictly applies only with no wind. With any wind the vector sum of wind due to motion of the bicycle plus true wind is to be taken instead of v. All in all this leads to the following equation:

$$P = m \cdot g \cdot (c_r + s) \cdot v + \frac{1}{2} K \cdot v^3. \quad (7)$$

3 Final time

The first task of the project was to compute a final time to finish the race for a certain power distribution. To do so the track was divided into small intervals to get a polygonal approximation of it. For each interval distances and slopes were computed using the theorem of Pythagoras. With these data equation (7) could be used to compute the velocity for a certain power level in every interval. The next step was to calculate the time for each interval, using equation (9) and sum everything up to gain the final time with equation (8).

$$\begin{aligned} T &= \sum_{1}^{N-1} t_i & (8)\\ t_i &= \frac{d_i}{v_i} & (9) \end{aligned}$$

Here T represents the final time and for each part of the track t_i, d_i, v_i are time, distance and velocity, respectively.
To get an idea in which parts of the track the velocity changes most depending on different power levels we run the simulation with 3 different power distributions. Therefore a constant power over the whole track was used to compute the finishing time:

$$P_1{=}550\ W, \quad P_2{=}435\ W, \quad P_3{=}300\ W$$

Figure (2) shows velocities for each power distribution:

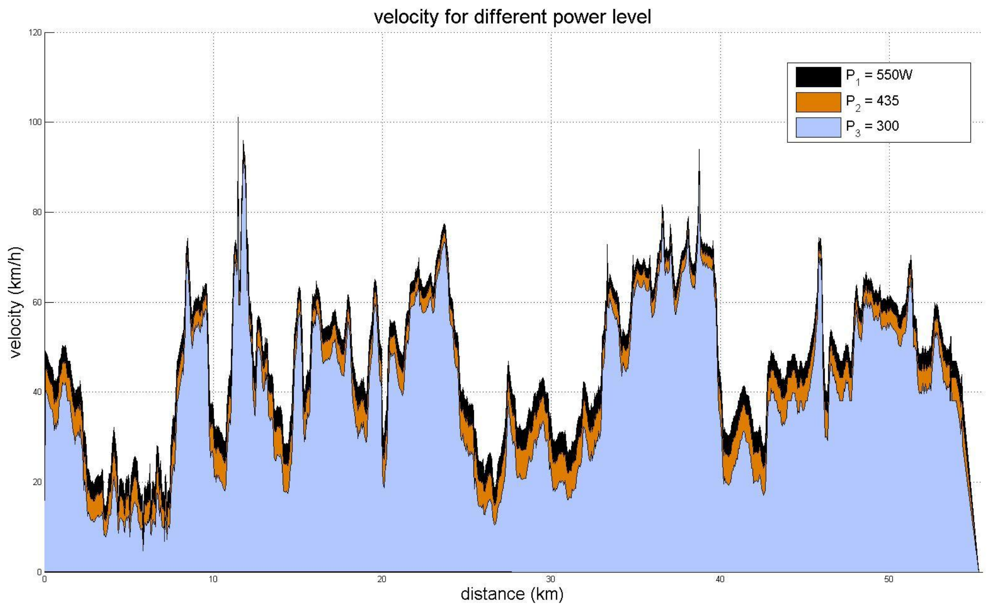


Figure 2: velocity for different power distributions

The finishing time was respectively:

$$T_1{=}1\text{h } 19\text{min } 5\text{sek}, \quad T_2{=}1\text{h } 29\text{min } 10\text{sek}, \quad T_3{=}1\text{h } 49\text{min } 50\text{sek}$$

As you can see from figure (2) the finishing time will obviously decrease, if the cyclist uses more power. But more interesting is, how much the velocity is affected with a higher power level in different parts. As you can see in the figure increasing the power in very steep parts of the track won't change the velocity much, the same can be observed going downhill. From this it is probably a good approach to increase power on "normal" slopes and use descents to recover energy. Although exhaustion isn't taken into account in this example P_2 is assumed as a power level a cyclist can easily maintain during the whole race. Therefore T_2 is the maximum time a cyclist needs to finish the given track.

4 The exhaustion constraint

The second task of the project was to compute an optimal power distribution to minimize the final time. In this section the model of exhaustion is used. It was already pointed out that a power level exists a cyclist can maintain without getting exhausted. Therefore using more power than this leads to exhaustion, which can't exceed a certain level. On the one hand the cyclist has to cover a fixed distance and he mustn't exceed his limits at any point during the race. Otherwise he would need to stop and rest. On the other hand the cyclist needs to be very close to his limits at the end of the race to use the available energy best. So the question is, how much power can be used in each interval without exceeding the limit and which parts profit most by given more power.
The exertion model used, is taken from the article [1] and it is based on the three-parameter critical power model:

$$D = \frac{A \cdot (P_m - P)}{(P_m - P_0) \cdot (P - P_0)}, \tag{10}$$

where D is the duration a cyclist can pedal at a power level P. P_m and P_0 are maximum and critical power level, respectively. The constant A is the anaerobic work capacity, which roughly defines the maximum amount of work a human can produce from the anaerobic energy system. Notice that using equation (10), puts restrains on the power outputs. This means the advisable power output should be between the maximum and critical power. For our project it is assumed that the cyclist has an advanced training level and the maximum and critical power outputs are the following:

$$P_m{=}1234\ W, \quad P_0{=}435\ W, \quad A{=}1.243\ \cdot 10^4 J$$

In defining the exertion, it is assumed that exertion increases linearly to the point of exhaustion and the rate of exertion is defined the reciprocal of equation (10):

$$R(P) = \frac{(P_m - P_0)(P - P_0)}{A(P_m - P)}. \tag{11}$$

The exertion E is then:

$$E(x) = \int_0^t R\ d\tau \tag{12}$$

at any point x of the course. The final exertion at the end of the race is

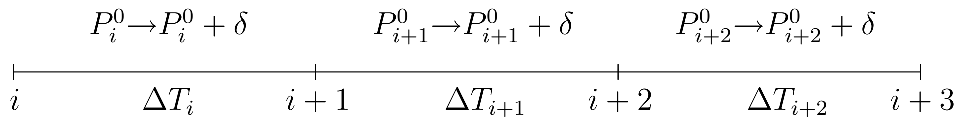
$$E(L) = \int_0^L \frac{R}{v}\ dx, \tag{13}$$

where L is the length of the whole course. Moreover it is assumed that exhaustion is restricted. The upper boundary is one and if the cyclist reaches this limit he must rest and can't continue the race. The lower boundary is set to zero, which means the cyclist is in full energy, without any sign of exhaustion. The cyclist can never gain more than full energy at any point. So mathematically the exhaustion has to fulfill the following inequality:

$$0 \leq \quad E(x) \quad \leq 1. \tag{14}$$

5 The algorithm

Finding the optimal power distribution for a track that minimizes time is a very hard problem. Computing the time of each interval needs solving the roots of a third degree polynomial. Because we do not know the derivates, a gradient based optimisation algorithms cannot be used. Moreover, there are lots of local optima. This is why we decided to solve the problem using a heuristic algorithm, which increases the power in randomly chosen intervals until increasing power in any interval would cause the cyclist to get exhausted. For computing the power distribution, it was essential to know, how to distribute power between different intervals, for example in which interval power should be increased, considering at the same time the exertion level. To respect all conditions and assumptions, a probability function is used to model the power distribution: Graphically it looks like this:


$P_i^0 \rightarrow P_i^0 + \delta$
$P_{i+1}^0 \rightarrow P_{i+1}^0 + \delta$
$P_{i+2}^0 \rightarrow P_{i+2}^0 + \delta$
i
ΔT_i
$i+1$
ΔT_{i+1}
$i+2$
ΔT_{i+2}
$i+3$

So the algorithm computes the distribution D, based on the time savings for each part and increases the interval that profits most by increased power.
The main idea of our heuristic algorithm follows the steps given below:

1. Initialise power distribution $P = (P_1, \cdots, P_N)$

2. Generate probability distribution D by calculating $\forall\ i \in \Omega$ equation (15)

$$D_i = \frac{T_i(P) - T_i(P+\delta)}{\sum_{k\in\Omega} T_k(P) - T_k(P+\delta)} \tag{15}$$

3. Select an interval $I \in \Omega$ with probabilities D

4. Increase the power P_I by ΔP

5. Check EXHAUSTION level by using equations (11) and (12), then
   - if exhausted for $P' = (P_1, \cdots, P_I + \Delta P, \cdots, P_N)$ then remove I from Ω.
   - if not exhausted set P=P'

6. If $\Omega = \emptyset$ then RETURN P, else continue from step 2

This algorithm is stochastic since the 3. step is randomized. To get a better result for the problem we do n loops of this algorithm and choose the best solution. More detailed description can be found in the pseudo code 5.1 and in the actual MATLAB-code, which is attached in A.

Algorithm 5.1: Pseudo code of used heuristic algorithm

```
for $i$ = $1..n$ do
    Tbest = $\infty$, Pbest = $\emptyset$ ;
    P = initial P distribution;
    while not Exhausted do
        calculate $D_i, \forall i \in \Omega$;
        select interval $I \in \Omega$ with probabilities D;
        $P_I = P_I + \Delta P$;
        compute $t_I$;
        if isExhausted(P,t) then
            $I \notin \Omega$;
            if $\Omega == \emptyset$ then
                $Exhausted = 1$;
                $T = \sum t_i$;
                if $T <$ Tbest then
                    $Tbest = T$;
                    Pbest = P;
                end
            end
        end
    end
end
```

6 Results

Using this idea the best solution that was obtained over almost hundred simulations is the following:

T=1h 28min 5s v$_{avg}$= 37km/h
P$_{avg}$=415 W E=0.9687

As you can see from the results, the cyclist has indeed a very good training level. The finishing time is quite well, considering that a hobby level cyclist would finish the track approximately around two hours. The average speed is $37km/h$, which is reasonable, because going downhill the speed of the cyclist can reach up to $100km/h$. The average power output is just below the critical power and the exhaustion at the end of the race is nearly one, which means the cyclist used almost all energy available.
In figures (3) and (4) you can see the graphical results for the optimal power distribution that was obtained through our simulations as well as corresponding velocity and exhaustion rate over the track.

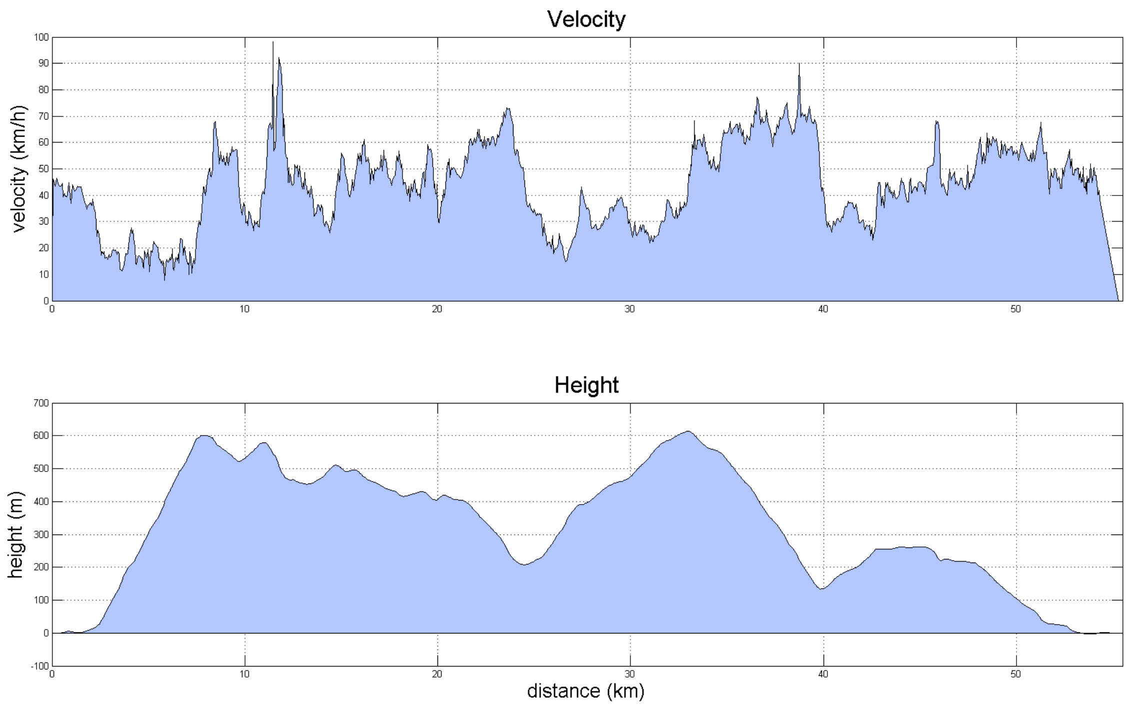


Figure 3: velocity for "optimal" power distribution and track

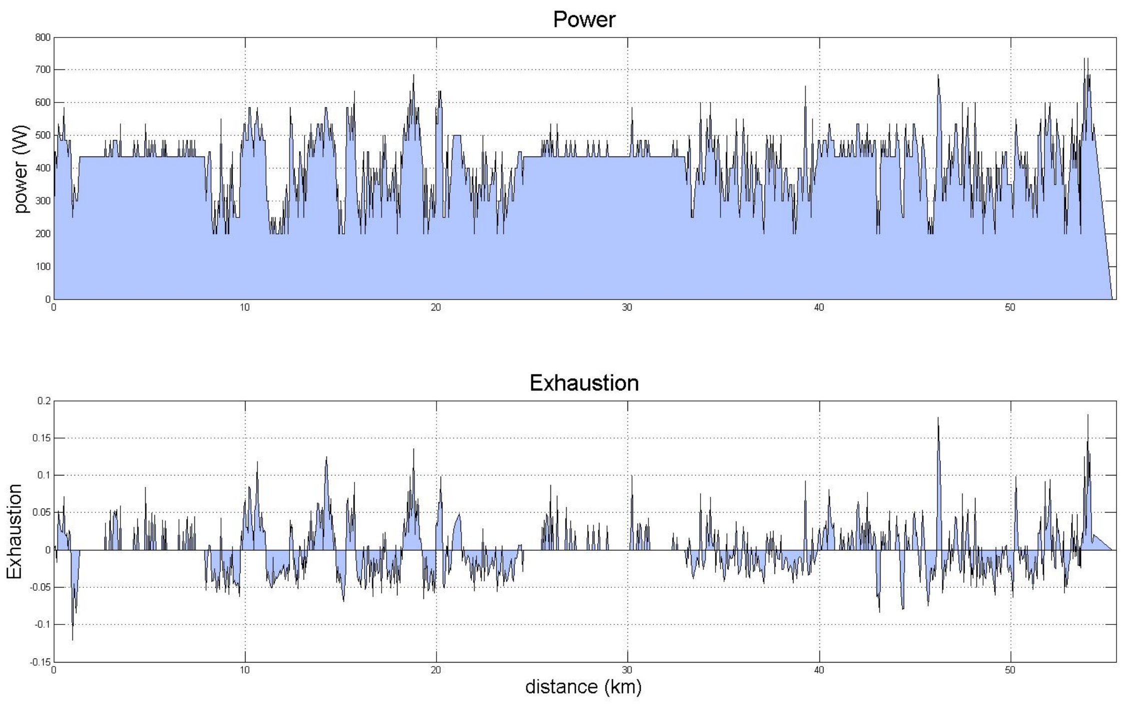


Figure 4: power distribution and rate of exhaustion over the track

7 Summary and future study

So all in all the first thing we had to do, was to formulate a mathematical model for the given problem. To do so, it was necessary to look into the fundamental physic equations, which gave us a reasonable idea of what we had to take into account. The next step was to find a connection between the total-time distribution on the one hand and the exhaustion-power distribution on the other hand. For reasons given in section (5), we could use the stochastic algorithm to find solutions for our problem. The results we obtained from the computer simulations were quite surprising. The best time we achieved, was 1h 28min 5sec, which is not much better than the time we got, assuming the cyclist moves with a constant power distribution, which was 1h 29min 10sec. The improvement is just about 1%.
Given these results we concluded that a constant power distribution equal to the critical power level a cyclist can maintain, depending on the skill level, results in a final time very close to the optimal time that can be achieved.
This means, during a race cyclist should always pedal at their critical power level and just occasionally - on short proper parts of the track - exceed it, if they have the opportunity to recover their energy afterwards. Of course in modern sports every second matters and can decide between winning and loosing a race, so our model could be a base for future research.
To improve it the following special cases could be considered:

1. we take into account acceleration and see how this affects the results
2. a wind resistance greater than zero could be modeled as well
3. the model to describe the exhaustion level could take for example the heart rate into account, which would probably give more realistic results

References

[1] *Scott Gordon, Optimising distribution of power during a cycling time trial, Sports Engineering, Volume 8, Number 2, December 2005, pp: 81-90, http://www.springerlink.com/content/ng9007744p4gl4j3/*

[2] *http://www.sheldonbrown.com/rinard/aero/formulas.htm*

[3] *David P. Swain: Cycling: Uphill and Downhill, Sportscience 2(4), 1998, http://www.sportsci.org/jour/9804/dps.html*

A MATLAB codes

```
function [n, d, angles, h] = calculate_trackdata(points)
% Calculates number of intervals (n), distances of points (d),
% angles between two points (angles) and height differences
% between two points.
% 'points' are assumed to be matrix where each row consists of
% one point where two first elements are coordinates of the
% point and third is the height of the point.

n_points = length(points(:,1));
n = n_points - 1;

%distances
d = zeros(n,1);
angles = d;
h = d;
for i = 2:(n_points)
    d(i-1) = norm(points(i,:)-points(i-1,:));
    h(i-1) = (-points(i,3)+points(i-1,3));
    angles(i-1) = asin(-h(i-1)/d(i-1));
end

if ~isempty(find(d == 0))
    warning('there was 0 distances at')
end

%--------------------------------------------------

function v = vel_i(P, m, cv, s, K)
% Computes the velocity in one interval with given P

g = 9.81;

p = [K/2 0 m*g*(cv+s) -P];
r = roots(p);

for i = 1:3
    if ~isreal(r(i))
        r(i) = -1;
    end
end
r(r<0) = [];

if length(r) ~= 1
    p
    error('no unique value for v');
end

v = r;

%--------------------------------------------------

function [t, v] = calculate_data(P, m, cv, K, n, d, angles)
% Computes the velosities (v) and times (t) in each interval with given power
% distribution (P)

% velocities
v = zeros(n,1);
for i = 1:n
    v(i) = vel_i(P(i), m, cv, tan(angles(i)), K);
    if isnan(v(i)) || ~isreal(v(i))
        t = [];
        return
    end
end

t = d./v;

%----------------------------------------------------

function bool = isExhaustion(P, P0, Pm, t)
% Checks from the power distribution if the cyclist is
% exhausted.

KR = 1.243e4;
R = (Pm-P0)*(P-P0)./(KR*(Pm-P));
E = t.*R;
n = length(E);

for i = 1:n
    if any(cumsum(E(i:n)) > 1)
        bool = 1;
        return
    else
        bool = 0;
    end
end

%----------------------------------------------------

function [T, Pbest] = distribute_power(P, deltaP, points, m, cv, K, v0, P0, Pm,
nloops, seed, Pd)
% Tries to find the best power distribution using heuristic methods.
% input:
% P initial power distribution
% deltaP
% points
% cv, K, v0, P0, Pm parameters
% nloops number of loops
% seed seed number for random number generator
% Pd initial power for intervals going down

T = inf;
Pbest = [];
[n, d, angles, h] = calculate_trackdata(points);
P(angles < 0) = Pd;
Pinit = P;
rand('state', seed);
```

```
% each loop tries to find a good solution
% best one of those is chosen
for i = 1:nloops
    P = Pinit;
    exhausted = 0;
    t_old = nan;
    [t, v] = calculate_data(P, m, cv, K, n, d, angles);
    allowedPoints = 1:n;

    while ~exhausted
       % calculate prob. dist.
       t2 = calculate_data(P+deltaP, m, cv, K, n, d, angles);
       allowP=sum(t(allowedPoints)-t2(allowedPoints));
       probs = (t(allowedPoints)-t2(allowedPoints))/allowP;
       cumprobs = cumsum(probs);

       % select an interval I
       r = rand; % random number from (0,1)
       P2big = [];
       for k = 1:length(allowedPoints)
           if r > cumprobs(k)
              continue
           else
              % add power to interval I
              I = allowedPoints(k);
              if P(I)+deltaP+0.05*Pm > Pm
                  P2big(end+1) = I;
                  continue
              end
              t_old = t;
              P(I) = P(I) + deltaP;
              v(I) = vel_i(P(I), m, cv, tan(angles(I)), K);
              t(I) = d(I)/v(I);
              break
           end
       end
       allowedPoints(P2big) = [];

       % check exchustion
       if isExhaustion(P, P0, Pm, t)
           allowedPoints(allowedPoints == I) = [];

           %fprintf('allowed points left: %d \n', length(allowedPoints));
           if isempty(allowedPoints)
              disp('it''s exhausted');
              fprintf('time : %d \n', sum(t));
              exhausted = 1;
           end
           % check if better solution
           P(I) = P(I) - deltaP;
           t = t_old;

           if sum(t) < T
               Pbest = P;
               T = sum(t);
```

```
            end
        else
            if all( (P+deltaP+0.05*Pm) > Pm)
                exhausted = 1;
                warning('Upper limit reached without exhaustion');
            end
        end
    end
end
```