Obliczenia z wykorzystaniem sztucznej inteligencji

ćwiczenia I

Metoda największego wzrostu

Symulowane wyżarzanie

Joanna Kołodziejczyk

03 marzec 2014

Plan

Prosty hill climbing

$s \leftarrow s_0$
$eval \leftarrow f(s)$
$k \leftarrow 0$
while $k < k_{max}$ **do**
  $s_n \leftarrow neighbour(s)$
  $eval_n \leftarrow f(s_n)$
  if $eval_n > eval$ **then**
    $s \leftarrow s_n$
    $eval \leftarrow eval_n$
  end if
  $k \leftarrow k + 1$
end while
return s

Steepest hill climbing

$s \leftarrow s_0; s_{best} \leftarrow s_0$
$eval \leftarrow f(s); eval_{best} \leftarrow f(s)$
loop
  $s_{list} \leftarrow neighbours(s)$
  for all x in s_{list} **do**
    $eval_n \leftarrow f(x)$
    if $eval_n > eval_{best}$ **then**
      $s_{best} \leftarrow x$
      $eval_{best} \leftarrow eval_n$
    end if
  end for
  if $eval_{best} \leq eval$ **then**
    return s
  end if
  $s \leftarrow s_{best}; eval \leftarrow eval_{best}$
end loop

Co musi być dane przed uruchomieniem algorytmu

- problem optymalizacyjny, ze zdefiniowaną, minimalizowaną funkcją celu f
- neighbour() lub neighbours() — funkcja generowania sąsiada lub wszystkich sąsiadów stanu bieżącego, która powinna wprowadzać małe losowe zmiany i być tak skonstruowana, by każde możliwe rozwiązanie było osiągalne
- ograniczenie liczby wykonanych kroków k_{max}

Plan

Algorytm

$s \leftarrow s_0$
$e \leftarrow E(s)$
$k \leftarrow 0$
$T_0 = tempestimation(s, E)$
while $k < k_{max}$ and $e > e_{max}$ **do**
  $s_n \leftarrow neighbour(s)$
  $e_n \leftarrow E(s_n)$
  if $P(e, e_n, temp(k)) > random()$ **then**
    $s \leftarrow s_n$
    $e \leftarrow e_n$
  end if
  $k \leftarrow k + 1$
end while
return s

Co musi być dane przed uruchomieniem algorytmu

- problem optymalizacyjny, ze zdefiniowaną, minimalizowaną funkcją celu E
- neighbour() — funkcja generowania sąsiada, która powinna wprowadzać małe losowe zmiany i być tak skonstruowana, by każde możliwe rozwiązanie było osiągalne
- $P(e, e_n, temp())$ — rozkład prawdopodobieństwa dla energii stanów s i s_n oraz temperatury $temp$
- $temp(iteracja)$ — schemat schładzania
- temperatura początkowa.

Przykładowy zestaw danych dla problemu komiwojażera

- problem komiwojażera: stan reprezentowany przez ciąg symbolizujący odwiedzane kolejno miasta $(m_1, m_2, \ldots, m_n)$, funkcją celu $E = \sum_{i=1}^{n-1} distance(m_i, m_{i+1}) + distance(m_n, m_1)$
- neighbour() — zamień losowo parę miast w ciągu.
- $P(e, e_n, temp()) = \begin{cases} e^{\frac{e-e_n}{temp()}}, & \text{jeżeli } e - e_n < 0 \\ 1, & \text{w przeciwnym wypadku} \end{cases}$
- $temp(iteracja)$ — schemat schładzania $T_{k+1} = \alpha * T_k$, gdzie $\alpha = 0.95$. Najprościej zmniejszać temperaturę w każdej iteracji.
- $T_0 = \sigma_0$: uruchamia się proces generowania stanów sąsiednich i oblicza się odchylenie standardowe σ_0 zmian funkcji celu.

Przykładowy zestaw danych dla optymalizacji funkcji dwóch zmiennych ciągłych

- problem optymalizacja f. dwóch zmiennych: stan reprezentowany przez parę zmiennych (x_1, x_2), funkcją celu E wzór na optymalizowaną funkcję
- neighbour() — zmień losowo zmienną $x_{k+1} = x_k + Cu$, gdzie u wektor losowych wartości znormalizowanych z zakresu $[-1, 1]$, a C stała macierz definiująca maksymalną zmianę każdej ze zmiennych.
- $P(e, e_n, temp()) = \begin{cases} e^{\frac{e-e_n}{temp()}}, & \text{jeżeli } e - e_n < 0 \\ 1, & \text{w przeciwnym wypadku} \end{cases}$
- $temp(iteracja)$ — schemat schładzania $T_{k+1} = \alpha * T_k$, gdzie $\alpha = 0.95$
- $T_0 = \sigma_0$: uruchamia się proces generowania stanów sąsiednich i oblicza się odchylenie standardowe σ_0 zmian funkcji celu.