A NEURAL NETWORK FOR DATA ASSOCIATION

Michel Winter and Gerard Favier

Laboratoire I3S CNRS UNSA
2000, route des Lucioles
Les Algorithmes, Bât. Euclide
06410 Sophia-Antipolis
FRANCE
e-mail: winter@i3s.unice.fr

ABSTRACT

This paper presents a new neural solution for solving the data association problem. This problem, also known as the multidimensional assignment problem, arises in data fusion systems like radar and sonar and targets tracking... Since it leads to an NP-complete combinatorial optimization, the optimal solution can not be reached in an acceptable calculation time, and the use of approximation methods like the Lagrangian relaxation is necessary.

In this paper, we propose an alternative approach based on a Hopfield neural model. We show that it converges to an interesting solution that respects the constraints of the association problem.

Some simulation results are presented to illustrate the behaviour of the proposed neural solution for an artificial association problem.

1. INTRODUCTION

The multidimensional assignment problem typically arises when several sensors observe the same environment. The data association process have to find which data correspond to the same object. The associations are generally decided in order to minimize a global criterion, using a kind of distance between the data of each sensor. The optimization of this criterion is the aim of this article. This problem also corresponds to an equivalent situation: a sensor observes the environment at several distinct moments. The association problem then consists in finding which data of each scan are generated by the same object. This situation essentially groups sonar and radar targets tracking [7].

If only two sensors are used, some algorithms [1] find the optimal solution in a acceptable calculation time. But when more than two sensors are considered, the problem becomes NP-complete and the optimal solution is not reachable even for a small amount of data. An approximation method is then necessary. Lagrangian relaxation method is generally chosen [2, 3, 4], but we show in this article that a Hopfield neural network is an interesting alternative that delivers an near optimal solution that respect the constraints of the optimization problem.

In the next section, we present the problem formulation and we describe the criterion to be optimized. Section 3 proposes our neural solution to the association problem. Some simulation results are presented in section 4 and concluding remarks are made in section 5.

2. PROBLEM FORMULATION

We define the following notations:

- $n$ the number of sensors (or the number of scans),
- $z_{ij}^l$ $i,j \in \{0,\ldots,m\}$, $j \in \{1,\ldots,n\}$ the $ij$-th data of the $j$-th sensor. We also introduce a dummy data $z_0^l$ that will encode the fact that no data from sensor $j$ is used,
- $Z$ the set of all real data, i.e. excepted dummy data $z_0^l$,
- $Z_{i_1,\ldots,i_n} = \{z_{i_1}^1, z_{i_2}^2, \ldots, z_{i_n}^n\}$ a data association,
- $\Gamma = \{Z_{i_1,\ldots,i_n}\}$ a partitioning of the data.

A partitioning $\Gamma$ is submitted to the following constraints:

(i) a data cannot be used twice:

$$\bigcap_{z_{i_1,\ldots,i_n} \in \Gamma} (Z_{i_1,\ldots,i_n} \cap Z) = \emptyset$$  (1)

We have to consider the intersection $Z_{i_1,\ldots,i_n} \cap Z$ because a dummy association can be used twice.

(ii) each data must be used exactly once:

$$\bigcup_{z_{i_1,\ldots,i_n} \in \Gamma} (Z_{i_1,\ldots,i_n} \cap Z) = Z$$  (2)

A cost $c_{i_1,\ldots,i_n}$ is also associated to each association $Z_{i_1,\ldots,i_n}$. The goal of the multidimensional assignment process is to find the partitioning $\Gamma^*$ so that:

$$\Gamma^* = \arg\min_{\Gamma \in \Omega} \sum_{z_{i_1,\ldots,i_n} \in \Gamma} c_{i_1,\ldots,i_n}$$  (3)

where $\Omega$ is the set of all possible partitioning $\Gamma$, i.e. $\Omega = \{\Gamma\}$.

If we define:

$$\rho_{i_1,\ldots,i_n} = \begin{cases} 1 & \text{if } Z_{i_1,\ldots,i_n} \in \Gamma^* \\ 0 & \text{otherwise} \end{cases}$$  (4)

[2, 3, 4]:

In this paper, we propose an alternative approach based on a Hopfield neural model. We show that it converges to an interesting solution that respects the constraints of the association problem. Some simulation results are presented to illustrate the behaviour of the proposed neural solution for an artificial association problem.
The search of partitioning \( \Gamma^* \) is equivalent to finding the values of the binary variables \( \rho_{i_1 \ldots i_n} \) that minimize:

\[
\Phi = \sum_{i_1=0}^{m_1} \ldots \sum_{i_n=0}^{m_n} \rho_{i_1 \ldots i_n} c_{i_1 \ldots i_n}
\]  

under the constraints:

\[
\begin{align*}
\sum_{i_k=0}^{m_k} \ldots \sum_{i_n=0}^{m_n} \rho_{i_1 \ldots i_n} & = 1 & \forall i_1 \in \{1 \ldots m_1\} \\
\sum_{i_1=0}^{m_1} \ldots \sum_{i_n=0}^{m_n} \rho_{i_1 \ldots i_n} & = 1 & \forall i_2 \in \{1 \ldots m_2\} \\
& \vdots & \\
\sum_{i_1=0}^{m_1} \ldots \sum_{i_{n-1}=0}^{m_{n-1}} \rho_{i_1 \ldots i_n} & = 1 & \forall i_n \in \{1 \ldots m_n\}
\end{align*}
\]  

another expression of the constraints can be: \( \forall \rho_{i_1 \ldots i_n} \)

\[
\rho_{i_1 \ldots i_n} = 1 \iff \rho_{i_1 \ldots i_n} + \rho_{i_j, i_1 \ldots i_{j-1} i_{j+1} \ldots i_n} \left(1 - \delta_{ij, 0}\right) + \ldots + \left(1 - \delta_{i_n, 0}\right) \rho_{i_1 \ldots i_{n-1} i_n} = 0
\]

where \( \delta_{ij} \) is the Kronecker symbol:

\[
\delta_{ij} = \begin{cases} 
1 & \text{if } i = j \\
0 & \text{otherwise}
\end{cases}
\]

Equation 7 can be rewritten as:

\[
\sum_{i_1=0}^{m_1} \ldots \sum_{i_{n-1}=0}^{m_{n-1}} \delta_{i_j i_1} \left(1 - \delta_{i_j, 0}\right) \rho_{i_1 \ldots i_{n-1} i_n} = 0
\]

3. NEURAL OPTIMIZATION

We propose to develop a Hopfield neural network that approximates a near optimal solution of the minimization of criterion 5 under constraints 9. The network minimizes the following energy:

\[
E_H = \sum_{i_1=0}^{m_1} \ldots \sum_{i_n=0}^{m_n} c_{i_1 \ldots i_n} f(x_{i_1 \ldots i_n})
\]

where \( f \) is the sigmoid function:

\[
f(x) = \frac{1}{1 + e^{-px}}
\]

where \( p \) is the slope of the function. The weights \( W_{i_j \ldots i_n} \) are given by:

\[
W_{i_j \ldots i_n} = C \sum_{j=1}^{n} \left[ \delta_{i_j i_j} \left(1 - \delta_{i_j, 0}\right) \left(1 - \prod_{k=1}^{n} \delta_{i_k i_k} \right) \right]
\]

The first term of energy \( E_H \) clearly corresponds to the minimization of the criterion 5. The second term is associated to the constraints 9. The last term is necessary to ensure that the network converges properly, but its influence is generally neglected by considering that the sigmoid function is sharp enough.

In the discrete time case, the neuron update rule that ensures the minimization of \( E_H \) is given by [5]:

\[
x_{i_1 \ldots i_n}(k) = -c_{i_1 \ldots i_n} - \sum_{i_1=0}^{m_1} \ldots \sum_{i_n=0}^{m_n} W_{i_1 \ldots i_n} f(x_{i_1 \ldots i_n}(k-1))
\]

We are now going to prove that the solution obtained at the convergence verifies the condition 7, under the condition:

\[
-1 < c_{i_1 \ldots i_n} < 0
\]

It is shown in appendix A that it is always possible to transform the costs \( c_{i_1 \ldots i_n} \) so that they verify 14, without changing the optimization process; thus, we will consider in the following that 14 is respected.

**Theorem 1:** neurons states \( x_{i_1 \ldots i_n} \) converge.

**Proof:** we know [6] that neurons outputs converge, i.e.:

\[
\lim_{k \to \infty} f(x_{i_1 \ldots i_n}(k)) = y_{i_1 \ldots i_n}^0, \forall i_1, \ldots, i_n
\]

where \( y_{i_1 \ldots i_n}^0 \) represents the output of the \( i_1 \ldots i_n \)-th neuron, at convergence. Using 13, we can write:

\[
\lim_{k \to \infty} x_{i_1 \ldots i_n}(k)
\]
where \( x^0_{i_1 \ldots i_n} \) is the neuron state at convergence.

**Theorem 2:** neurons outputs converge to values in \( \{0, 0.5, 1\} \) when the slope \( p \) of the sigmoid function tends toward infinity.

**Proof:** we have:

\[
y^0_{i_1 \ldots i_n} = f(x^0_{i_1 \ldots i_n}) = \frac{1}{1 + e^{-4p x^0_{i_1 \ldots i_n}}} \] (17)

and:

\[
\lim_{p \to \infty} f(x) = \begin{cases} 
0 & \text{if } x < 0 \\
0.5 & \text{if } x = 0 \\
1 & \text{if } x > 0 
\end{cases} 
\] (18)

using 17 and 18, proof is evident.

**Theorem 3:** neurons outputs converge to values in \( \{0, 1\} \) if \( C > 2 \).

**Proof:** we have to show that the case \( x^0_{i_1 \ldots i_n} = 0 \) is impossible. At convergence, 13 can be rewritten as follows:

\[
x^0_{i_1 \ldots i_n} = -c_{i_1 \ldots i_n} - \frac{\alpha C}{2} = 0 \quad \text{with } \alpha \in \mathbb{N}^+ \] (20)

So, if \( C > 2 \) it is easy to see that \( x^0_{i_1 \ldots i_n} = 0 \) is impossible.

**Theorem 4:** solution given by the neural network at convergence verifies the constraint 7.

**Proof:** we denote \( A \) the proposition "\( y^0_{i_1 \ldots i_n} = 1 \)“, and \( B \) the proposition:

\[
\sum_{i_1^0 = 0}^{m_1} \ldots \sum_{i_n^0 = 0}^{m_n} \delta_{i_1 \ldots i_n} (1 - \delta_{i_1 \ldots i_n}) (1 - \prod_{i_j \in \mathbb{W}} \delta_{i_j}) y^0_{i_1 \ldots i_n} = 0 \] (21)

Constraint 7 is verified if we have:

\[
A \iff B \] (22)

or:

\[
B \Rightarrow A \] (23)

and \( \overline{B} \Rightarrow \overline{A} \) (24)

where \( \overline{A} \) represents the proposition \( y^0_{i_1 \ldots i_n} = 0 \), and \( \overline{B} \) is:

\[
\sum_{i_1^0 = 0}^{m_1} \ldots \sum_{i_n^0 = 0}^{m_n} \delta_{i_1 \ldots i_n} (1 - \delta_{i_1 \ldots i_n}) (1 - \prod_{i_j \in \mathbb{W}} \delta_{i_j}) y^0_{i_1 \ldots i_n} > 0 \] (25)

First, we show that \( B \Rightarrow A \):

\[
B \iff \sum_{i_1^0 = 0}^{m_1} \ldots \sum_{i_n^0 = 0}^{m_n} \delta_{i_1 \ldots i_n} (1 - \delta_{i_1 \ldots i_n}) (1 - \prod_{i_j \in \mathbb{W}} \delta_{i_j}) y^0_{i_1 \ldots i_n} = 0
\]

Multiplying by a constant \( C \neq 0 \) leads to:

\[
B \Rightarrow \sum_{i_1^0 = 0}^{m_1} \ldots \sum_{i_n^0 = 0}^{m_n} C \delta_{i_1 \ldots i_n} (1 - \delta_{i_1 \ldots i_n}) (1 - \prod_{i_j \in \mathbb{W}} \delta_{i_j}) y^0_{i_1 \ldots i_n} = 0
\] (26)

using 19 and 14, this leads to:

\[
B \Rightarrow x^0_{i_1 \ldots i_n} = -c_{i_1 \ldots i_n} > 0
\] (27)

and:

\[
B \Rightarrow y^0_{i_1 \ldots i_n} = 1
\] (28)

showing that \( B \Rightarrow A \). We must now verify that \( \overline{B} \Rightarrow \overline{A} \):

\[
\overline{B} \iff \sum_{i_1^0 = 0}^{m_1} \ldots \sum_{i_n^0 = 0}^{m_n} \delta_{i_1 \ldots i_n} (1 - \delta_{i_1 \ldots i_n}) (1 - \prod_{i_j \in \mathbb{W}} \delta_{i_j}) y^0_{i_1 \ldots i_n} > 0
\]

We proved that \( y^0_{i_1 \ldots i_n} \in \{0, 1\} \); so, multiplying by a constant \( C \neq 0 \) leads to:

\[
\overline{B} \Rightarrow \sum_{i_1^0 = 0}^{m_1} \ldots \sum_{i_n^0 = 0}^{m_n} C \delta_{i_1 \ldots i_n} (1 - \delta_{i_1 \ldots i_n}) (1 - \prod_{i_j \in \mathbb{W}} \delta_{i_j}) y^0_{i_1 \ldots i_n} = C
\]

adding \( c_{i_1 \ldots i_n} \) and multiplying by \( -1 \) show that:

\[
\overline{B} \Rightarrow x^0_{i_1 \ldots i_n} < 0
\] (29)

\[
y^0_{i_1 \ldots i_n} = 0
\] (30)

which shows that \( \overline{B} \Rightarrow \overline{A} \).
4. SIMULATIONS RESULTS

We made some simulations in order to examine the behaviour of the proposed neural method. We consider four sensors delivering points in the 2 dimensional plane. The goal of the data association is to form figures with 2, 3 or 4 points, each figure containing at most one data of each sensor. The cost of each association is the perimeter of the figure divided by its number of segments (i.e. 1 segment for a 2 points association, 3 segments for a 3 points association and 4 segments for a 4 points association). We have generated 50 different association problems and we have computed the optimal solution for each of them. 17.4% of the solutions given by our network are optimal ones, and the mean error between the optimal solution and the obtained solution equals 2.7%. 100% of the obtained solutions verify the constraints.

5. CONCLUSION

We proposed a neural approach to solve the data association problem that arises in data fusion systems or in radar [7] and sonar [2] tracking. We proved that the obtained solution always verifies the constraints of the problem, and some simulations showed that our approach always converges to a near optimal solution.

6. REFERENCES


A. COSTS NORMALIZATION

We show in this section that it is possible to change the costs $c_{i_1...i_n}$ into the costs $C_{i_1...i_n} = c_{i_1...i_n} + \sum_{j=1}^{n} a_{ij}^l$, with $a_{ij}^l = 0 \ \forall j \in \{1,...,n\}$, without changing the optimization process. We define:

$$J = \sum_{i_1=0}^{m_1} \ldots \sum_{i_n=0}^{m_n} C_{i_1...i_n} \rho_{i_1...i_n}$$

Expression of criterion $J$ can be rewritten as follows:

$$J = \sum_{i_1=0}^{m_1} \ldots \sum_{i_n=0}^{m_n} \left( c_{i_1...i_n} + \sum_{j=1}^{n} a_{ij}^l \right) \rho_{i_1...i_n}$$

$$= \Phi + \sum_{i_1=0}^{m_1} \ldots \sum_{i_n=0}^{m_n} \left( \sum_{j=1}^{n} a_{ij}^l \right) \rho_{i_1...i_n}$$

We must show that the second term of 32 is constant. We have:

$$\sum_{i_2=0}^{m_2} \ldots \sum_{i_n=0}^{m_n} \rho_{i_1...i_n} = 1 \ \forall \ i_1 \in \{1,...,m_1\}$$

and so:

$$\sum_{i_2=0}^{m_2} \ldots \sum_{i_n=0}^{m_n} \rho_{i_1...i_n} a_{il}^l = a_{il}^l \ \forall \ i_1 \in \{1,...,m_1\}$$

We have $a_{il}^l = 0$, thus we can write:

$$\sum_{i_2=0}^{m_2} \ldots \sum_{i_n=0}^{m_n} \rho_{i_1...i_n} a_{il}^l = \sum_{i_2=0}^{m_2} \ldots \sum_{i_n=0}^{m_n} \rho_{i_1...i_n} a_{il}^l$$

taking into account each sensor, we have:

$$\left\{ \begin{array}{l} \sum_{i_2=0}^{m_2} \ldots \sum_{i_n=0}^{m_n} \rho_{i_1...i_n} a_{il}^l = \sum_{i_2=0}^{m_2} \ldots \sum_{i_n=0}^{m_n} a_{il}^l \\ \vdots \\ \sum_{i_2=0}^{m_2} \ldots \sum_{i_n=0}^{m_n} \rho_{i_1...i_n} a_{il}^n = \sum_{i_2=0}^{m_2} \ldots \sum_{i_n=0}^{m_n} a_{il}^n \end{array} \right.$$

and:

$$\sum_{i_2=0}^{m_2} \ldots \sum_{i_n=0}^{m_n} \left( \sum_{j=1}^{n} a_{ij}^l \right) \rho_{i_1...i_n} = \sum_{i_2=0}^{m_2} \ldots \sum_{i_n=0}^{m_n} \left( \sum_{j=1}^{n} a_{ij}^l \right)$$

which is a constant. This result shows that it is possible to add constants terms to each costs without changing the minimization process. By using this result and by dividing the costs by a normalization factor, it is always possible to consider costs that verify 14.