Managing Uncertain Temporal Relations using a Probabilistic Interval Algebra

Malek Mouhoub and Jia Liu
Computer Science Department, University of Regina
Regina, SK, Canada S4S 0A2
{mouhoubm,liujia204}@cs.uregina.ca

Abstract—We propose a probabilistic extension of Allen’s Interval Algebra for managing uncertain temporal relations. Although previous work on various uncertain forms of quantitative and qualitative temporal networks have been proposed in the literature, little has been addressed to the most obvious type of uncertainty, namely the probabilistic one. More precisely, our model adapts the probabilistic Constraint Satisfaction Problem (CSP) framework in order to handle uncertain symbolic temporal constraints. In a probabilistic CSP, each constraint C is given a probability of its existence in the real world. There is thus more than one CSP to solve as opposed to the traditional CSP where no such uncertainties exist. In a probabilistic temporal CSP, since we use the Interval Algebra where a constraint is a disjunction of Allen primitives, the probability is assigned to each of these Allen primitives rather than to the temporal constraint itself. This means that a probabilistic temporal CSP involves many possible temporal CSPs, each with a probability of its existence. Solving a probabilistic temporal CSP consists of finding a scenario that has the highest probability to be the solution for the real world. This is an optimization problem that we solve using a branch and bound algorithm we propose and involving constraint propagation. Experimental study conducted on randomly generated temporal problems demonstrates the efficiency in time of our solving method.

I. INTRODUCTION

A Constraint Satisfaction Problem (CSP) [1], [2], [3], [4] can be modeled as a tuple \( < X, D, C > \) where \( X \) is a set of variables, \( D \) the set of the variables domains and \( C \) the set of constraints restricting the values that the variables can simultaneously take. Solving a CSP consists of finding an assignment to each variable from a value of its domain such that all the constraints are satisfied. CSPs have been used in many real world applications including those under temporal constraints. In this latter case we talk about temporal constraint networks [5], [6], [7], [8]. In a temporal constraint network, variables, corresponding to temporal objects, are defined on a set of time points or time intervals while constraints can either restrict the domains of the variables and/or represent the relative position between variables. The relative position between variables can be expressed via qualitative or quantitative relations. Quantitative relations are temporal distances between temporal variables while qualitative relations represent incomplete and less specific symbolic information between variables. Constraint propagation techniques and backtrack search are then used to check the consistency of the temporal network and to infer new temporal information.

When solving real life applications under (temporal) constraints we often need to deal with uncertain information. These latter are the result of uncontrollable external factors. Variables, domains and constraints in CSPs and temporal CSPs can thus be uncertain and some research work in this regard has been addressed. In a Mixed-CSP [9], variables are divided into two categories. One includes controllable variables (decision variables) that are totally under control of users and the other includes uncontrollable variables (environmental variables) that are not under control. The uncertainty is reflected by uncontrollable variables. In the stochastic CSP [10], a probability distribution is associated with the domain of each variable. In probabilistic CSPs [9], the uncertain factors are the probabilities of the existence of various constraints. Simple Temporal Networks [7], very popular for representing quantitative temporal information, have been augmented to model the uncertain duration between time points. One of the resulting frameworks is called Simple Temporal Network with Uncertainty (STNU) [11]. Solving algorithms and tractable classes for the STNU have been proposed in [12], [13]. Another resulting framework is the Fuzzy Temporal Constraint Network [14]. This latter model extends the STN with fuzzy durations using the possibilistic distributions. Qualitative temporal networks handling uncertainty include the fuzzy extension of Allen’s Interval Algebra [15] and the Probabilistic Temporal Interval Network (PTI) [16]. Both of these two formalisms are based on the possibility theory which is used to model the uncertain symbolic relations by assigning a preference degree to every basic Allen’s primitive [5]. A path consistency algorithm has been proposed in each model.

In this paper we propose a new framework to manage uncertain symbolic temporal networks. More precisely, our model adapts the probabilistic CSP framework in order to handle uncertain symbolic temporal constraints. In a probabilistic CSP, each constraint C is given a probability of whether it exists or not in the real world. Since a symbolic temporal constraint is a disjunction of basic Allen primitives, in the probabilistic temporal CSP each basic primitive in the disjunctive temporal constraint is given a probability, rather than assigning a single probability to the whole constraint. We call a Probabilistic Interval Algebra (PIA) Network a IA network where primitives within each disjunctive relation have a probability of their existence. Solving a PIA network consists of finding a scenario that has the highest probability to be the...
solution for the real world. For that, we propose a branch and bound algorithm based on Path Consistency (PC) for finding this solution, called also robust solution. In order to evaluate the performance in time of our solving method, we conducted an experimental study on randomly generated PIA networks. The results demonstrate the efficiency in time of our method for dealing with large size problems.

The closest literature to our work are the PTI [16] and the Fuzzy Temporal Constraint Network [14]. While these two frameworks are based on possibility theory, our addresses the most obvious type of uncertainty, namely the probabilistic one. More precisely, in the PTI network [16] the probabilities of all Allen primitives within a given disjunctive relation sum to 1 (are normalized) which is not the case of our model where each primitive (within a given relation) has a probability between 0 and 1. Also we found that PTI is a particular case of CSPs with preferences or fuzzy CSPs. Indeed the composition in the PC algorithm is based on the max operator.

In the next section we present through examples our new framework handling uncertainties. Section 3 describes our branch and bound algorithm for finding the most robust solution. In section 4 we provide a description of the experiments we conducted on randomly generated temporal problems. We finally conclude in section 5 with some remarks and possible future works.

II. Probabilistic Symbolic Temporal Network

A. Interval Algebra (IA) Networks

An Interval Algebra Network (also called IA network) consists of a tuple $<E, SET_R>$ where $E$ is a set of events and $SET_R$ the set of binary constraints between events. A relation $R_{ij} \in SET_R$ represents the relative position between two events $e_i$ and $e_j$ and is expressed by the disjunction of some Allen primitives [5]. Table I lists all the Allen primitives. For instance the relation $R = M \lor O$ between two events $e_1$ and $e_2$ represents the fact that $e_1$ meets or overlaps the event $e_2$. Since there are 13 Allen primitives, the set $SET_R$ contains $2^{13}$ possible relations. One particular relation called universal relation (or identity relation) and denoted by $I$ corresponds to the disjunction of the 13 primitives. This relation expresses the fact that the relation between the two involved events is completely unknown. Solving an IA network consists of assigning to each disjunctive relation one of its primitives such that all the relations are consistent together.

In order to illustrate the Interval Algebra and its related IA network, let us consider the following example taken from [1].

Example 1: Fred was reading the paper while eating his breakfast. He put the paper down and drank the last of his coffee. After breakfast he went for a walk.

The above story can be represented by the IA network shown in figure 1. Here each node corresponds to a given event in the story and each edge is labeled by a binary constraint between events. For example, the constraint between the events Paper (reading paper) and Coffee (drinking coffee) is expressed by the disjunctive relation DOS ($D \lor O \lor S$). This relation means that between the events Paper and Coffee we can have one of the following three primitives: During, Overlaps or Starts. Note that, since there is no constraint between Coffee and Walk, we could represent the corresponding relation by the universal relation $I$.

A possible solution for the above temporal problem is given in figure 1 [1]. This solution corresponds to the following:

- Paper $O$ Coffee.
- Paper $D$ Breakfast.
- Coffee $D$ Breakfast.
- Breakfast $B$ Walk.

B. Probabilistic Symbolic Temporal Relations (PSTR)

In probabilistic CSPs [9], constraints can be certain or uncertain. Each uncertain constraint has a probability of its existence. For instance, $Pr(C) = 0.7$ means that the constraint $C$ (involving a list of variables) has 70% chances to exist.

In the case of IA networks, we associate these type of probabilities to the Allen primitives within a given uncertain relation rather than to the relation itself. More formally, we define a Probabilistic Symbolic Temporal Relation (PSTR) as follows.

Definition 1: Probabilistic Allen Primitive (PAP): A Probabilistic Allen Primitive (PAP) $r$ is an Allen primitive that has a probability of its existence within a relation $R_{ij}$ between two events $e_i$ and $e_j$. More precisely $r$ has the following probabilities where $0 \leq p \leq 1$.

- $Pr(r \in R_{ij}) = p$: the probability that $r$ exists within $R_{ij}$.
- $Pr(r \notin R_{ij}) = 1 - p$: the probability that $r$ does not exist in $R_{ij}$.

The PAP $r$ is certain (or completely known) if and only if $Pr(r \in R_{ij}) = 1.0$. In this case we simply call it Allen Primitive (AP).

<table>
<thead>
<tr>
<th>Relation</th>
<th>Symbol</th>
<th>Inverse</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X Before Y</td>
<td>B</td>
<td>Bi</td>
<td>X ______ Y</td>
</tr>
<tr>
<td>X Equals Y</td>
<td>E</td>
<td>E</td>
<td>X ______ Y</td>
</tr>
<tr>
<td>X Meets Y</td>
<td>M</td>
<td>Mi</td>
<td>X ______ Y</td>
</tr>
<tr>
<td>X Overlaps Y</td>
<td>O</td>
<td>Oi</td>
<td>X ______ Y</td>
</tr>
<tr>
<td>X During Y</td>
<td>D</td>
<td>Di</td>
<td>X ______ Y</td>
</tr>
<tr>
<td>X Starts Y</td>
<td>S</td>
<td>Si</td>
<td>X ______ Y</td>
</tr>
<tr>
<td>X Finishes Y</td>
<td>F</td>
<td>Fi</td>
<td>Y ______ X</td>
</tr>
</tbody>
</table>

TABLE I

ALLEN PRIMITIVES.
Definition 2: Probabilistic Symbolic Temporal Relation (PSTR): Given a relation \( R_{ij} \) between two events \( e_i \) and \( e_j \); and defined by the disjunction \( r_1 \vee r_2 \cdots \vee r_n \) where \( n \leq 13 \) and the \( r_s \) are Allen primitives. \( R \) is a Probabilistic Symbolic Temporal Relation (PSTR) if and only if at least one of its \( r_s \) is a PAP.

The PSTR \( R_{ij} \) is certain (or completely known) if and only if all its \( r_s \) are certain (APs). In this latter case \( R_{ij} \) is simply an STR. \( \Pr(r) \) means that we have one primitive (\( \Pr \) being the most probable one). Let us assign the following probabilities to each of these two primitives. \( \Pr(S) = 0.6 \) and \( \Pr(O) = 0.8 \). Using the above the PSTR \( R \) between \( \text{Paper} \) and \( \text{Coffee} \) is equal to one of three possible STRs as follows.

1) \( \Pr(R = S) = \Pr(S) \cdot (1 - \Pr(O)) = 0.6 \cdot 0.2 = 0.12 \)
2) \( \Pr(R = O) = \Pr(O) \cdot (1 - \Pr(S)) = 0.8 \cdot 0.4 = 0.32 \)
3) \( \Pr(R = O \vee S) = 0.6 \cdot 0.8 = 0.48 \)

The probability that \( R \) exists between \( \text{Paper} \) and \( \text{Coffee} \) is computed as follows.
\[
\Pr(R) = \Pr(R = O) + \Pr(R = S) + \Pr(R = O \vee S) = 0.92
\]

C. Probabilistic IA (PIA) Network

Definition 3: A Probabilistic IA (PIA) network is a IA network where some disjunctive relations are PSTRs.

According to the above new information, there are now only two primitives (instead of three as shown in Figure 1) between the events \( \text{Paper} \) and \( \text{Coffee} \). These primitives are \( O \) and \( S \) with \( O \) being the most probable one. Let us assign the following probabilities to each of these two primitives. \( \Pr(S) = 0.6 \) and \( \Pr(O) = 0.8 \). Using the above the PSTR \( R \) between \( \text{Paper} \) and \( \text{Coffee} \) is equal to one of three possible STRs as follows.

1) \( \Pr(R = S) = \Pr(S) \cdot (1 - \Pr(O)) = 0.6 \cdot 0.2 = 0.12 \)
2) \( \Pr(R = O) = \Pr(O) \cdot (1 - \Pr(S)) = 0.8 \cdot 0.4 = 0.32 \)
3) \( \Pr(R = O \vee S) = 0.6 \cdot 0.8 = 0.48 \)

The probability that \( R \) exists between \( \text{Paper} \) and \( \text{Coffee} \) is computed as follows.
\[
\Pr(R) = \Pr(R = O) + \Pr(R = S) + \Pr(R = O \vee S) = 0.92
\]

D. Probable Worlds and Robust Solutions

1) Definition 4: A probable world of a given PIA network \( P \) is a scenario (IA network) corresponding to

\[
\begin{align*}
&\text{Initial problem} \\
&\text{Solution}
\end{align*}
\]

Fig. 1. Interval Algebra (IA) Constraint Network [1].

Fig. 2. PIA network Corresponding to examples 1, 3 and 4.
replacing each PSTR of P by one of its possible STRs. A basic world is a probable world where all the selected STRs are APs.

Let us consider the PIA network in Figure 2 with two PSTRs: \((\text{Paper, Coffee})\) and \((\text{Breakfast, Walk})\). A probable world (respectively basic world) can be obtained by replacing each of these two PSTRs by one of their STRs (respectively APs). Below we have the list of the three possible probable worlds where the first two are basic worlds.

1. \(W_1\):
   - Paper \(O\) Coffee.
   - Paper \(E \lor D \lor D\lor O \lor O \lor S \lor S \lor F \lor F\) - Breakfast.
   - Coffee \(D\) Breakfast.
   - Breakfast \(B\) Walk.

2. \(W_2\):
   - Paper \(S\) Coffee.
   - Paper \(E \lor D \lor D\lor O \lor O \lor S \lor S \lor F \lor F\) - Breakfast.
   - Coffee \(D\) Breakfast.
   - Breakfast \(B\) Walk.

3. \(W_3\):
   - Paper \(O S\) Coffee.
   - Paper \(E \lor D \lor D\lor O \lor O \lor S \lor S \lor F \lor F\) - Breakfast.
   - Coffee \(D\) Breakfast.
   - Breakfast \(B\) Walk.

Note that, in the above, \(W_1\) (respectively \(W_2\)) is included in \(W_3\). Thus the set of solutions to \(W_1\) and \(W_2\) are included in the set of solutions to \(W_3\).

Example 5: Let us consider another PIA network with two PSTRs \(C_1\) and \(C_2\), and two STRs \(C_3\) and \(C_4\) defined as follows.
- \(C_1 = M \lor S\) where \(Pr(M) = 0.2\) and \(Pr(S) = 0.5\).
- \(C_2 = O \lor F\) where \(Pr(O) = 0.3\) and \(Pr(F) = 0.6\).
- \(C_3 = P\)
- \(C_4 = Pi \lor Oi\)

\(C_1\) and \(C_2\) have each 3 possible temporal relations. One probable world can be defined, for example, by the following constraints \(\{C_1 = M, C_2 = O \lor F, C_3 = P, C_4 = Pi \lor Oi\}\). The total number of probable worlds will thus be equal to 9.

Definition 5: The total number of possible worlds of a PIA network \(P\) can be computed as follows.

\[
\prod_{C_i \in P} \text{dim}(C_i)
\]

where \(C_i\) is a PSTR and \(\text{dim}(C_i)\) is the number of possible STRs that \(C_i\) has.

Definition 6: Let us consider a given PIA network \(P\) and \(W_5\) the set of probable worlds of \(P\) that can be satisfied by a given solution \(S\). The robustness of \(S\) is computed as follows.

\[
\text{Robustness}(S) = \sum_{W_i \in W_5} Pr(W_i)
\]

where, \(Pr(W_i)\), the probability of the world \(W_i\) is:

\[
Pr(W_i) = \prod_{C_j \in W_i} Pr(C_j)
\]

\(C_j\) is here a PSTR of the probable world \(W_i\).

Definition 7: The most robust solution to a given problem \(P\) is the solution that has the highest degree of robustness.

In other words, the above definition means that the most robust solution is the one that has the highest probability to satisfy the real world. Solving a PIA Network is an optimization problem that consists of finding the most robust solution.

III. FINDING THE MOST ROBUST SOLUTION

Example 6: Let us consider the PIA network of example 5. Here we have 4 basic worlds.

1. \(W_1 = \{C_1 = M, C_2 = O, C_3, C_4\}\)
2. \(W_2 = \{C_1 = M, C_2 = F, C_3, C_4\}\)
3. \(W_3 = \{C_1 = S, C_2 = O, C_3, C_4\}\)
4. \(W_4 = \{C_1 = S, C_2 = F, C_3, C_4\}\)

Notice that the solutions of the above four basic worlds are all the solutions of the corresponding PIA network. Thus, in order to search for the most robust solution in a PIA network we can simply look for the most robust solution in the basic worlds only.

The probability of a basic world is the product of the probabilities of all the uncertain primitives this basic world involves. For instance, the probability of each of the above four worlds is computed as follows.

1. \(Pr(W_1) = Pr(M) \times (1 - Pr(S)) \times Pr(O) \times (1 - Pr(F)) = 0.2 \times 0.5 \times 0.3 \times 0.4 = 0.012\)
2. \(Pr(W_2) = Pr(M) \times (1 - Pr(S)) \times Pr(F) \times (1 - Pr(O)) = 0.2 \times 0.5 \times 0.6 \times 0.7 = 0.042\)
3. \(Pr(W_3) = Pr(S) \times (1 - Pr(M)) \times Pr(O) \times (1 - Pr(F)) = 0.3 \times 0.8 \times 0.3 \times 0.4 = 0.048\)
4. \(Pr(W_4) = Pr(S) \times (1 - Pr(M)) \times Pr(F) \times (1 - Pr(O)) = 0.3 \times 0.8 \times 0.6 \times 0.7 = 0.16\)

The most probable basic world is the basic world with the most probable uncertain primitives. In the above example, \(W_4\) is the most probable world.

The robustness of a solution corresponds here to the probability of the basic world it satisfies. The most robust solution is the one that satisfies the most probable consistent world.

A. Solving Algorithm

Solving an IA network is an NP-hard problem that requires a backtrack search algorithm of exponential time cost [8]. In order to overcome this difficulty in practice, constraint propagation based on path consistency is used before and during the search in order to prevent earlier later failure. In the case of PIA networks we will also use backtrack search with path consistency performed before and during the search.

Note that, when path consistency is used before the search it is applied to hard (certain) constraints only. If the path consistency fails then the PIA network is not consistent and there is thus no need to proceed with the search. In the case where the path consistency is successful then the hard constraints of the resulting network will have less Allen primitives which will reduce the size of the search space.

More precisely, we propose the following branch and bound algorithm for finding the most robust solution.
1) Perform path consistency to the subnetwork containing only the hard constraints. If the subnetwork is not path consistent return that the IA network is not consistent. Figure 3 presents the pseudo-code of the path consistency algorithm [8].

2) For each uncertain constraint, sort its primitives by decreasing order of their probability.

3) Following the forward check principle [4], pick a constraint, assign to it one of its Allen primitives and run the path consistency algorithm on the subnetwork containing the newly assigned constraint and the non assigned ones. If path consistency fails then assign another primitive to the current constraint or backtrack to the previous assigned constraint if the current constraint does not have any another primitive to assign. If the path consistency is successful, select another constraint and redo the same process until all the constraints are assigned in which case we obtain a solution otherwise return that the PIA network is not consistent. If a solution is obtained, compute its robustness (the probability of the complete assignment it satisfies) and assign the result to the lower bound (LB).

4) The rest of the search space is then systematically explored as follows. Each time the current constraint is assigned a primitive, if this constraint is uncertain (PSTR) then an overestimation of the robustness of any possible solution following this decision is computed and used as an upper bound (UB). If UB < LB then the current uncertain constraint is assigned another value or backtrack to the previous constraint if all the primitives have been explored. The overestimated robustness is the product of the probabilities of all the assigned primitives plus the product of the max probabilities of the uncertain constraints that are not yet assigned. The max probability of a non assigned uncertain constraint corresponds to the largest probability of its primitives.

**Function** DPC(listOfHardConstraints)

1. PC ← false
2. L ← listOfHardConstraints
3. while (L ≠ 0) do
4.  select and delete an (x, y) from L
5.  for k ← 1 to n, k ≠ x and k ≠ y do
6.   t ← Csk ∩ Cxy, Cyk
7.   if (t ≠ Csk) then
8.     Csk ← t
9.     Cks ← INVERSE(t)
10.    L ← L ∪ {(x, k)}
11.   updated_list ← updated_list ∪ {(x, k)}
12.   t ← Cxy ∩ Cxy, Cyk
13.  if (t ≠ Cxy) then
14.   Cxy ← INVERSE(t)
15.   L ← L ∪ {(k, y)}
16.  updated_list ← updated_list ∪ {(y, k)}

Fig. 3. Path Consistency Algorithm

**IV. EXPERIMENTATION**

In order to evaluate the solving method we propose, we have performed several tests on randomly generated consistent PIA networks. The experiments are performed on a PC Pentium 4 computer under Windows XP system. All the procedures are coded in C++. A consistent IA network of size N (N is the number of variables) is randomly generated as follows. We first randomly generate a numeric solution (set of N numeric intervals), extract the Allen primitives that are consistent with the numeric solution and randomly add other primitives to get the set of constraints of the generated problem. To generate a PIA network having a percentage p of uncertain constraints, we randomly pick p*C constraints (where C is the total number of constraints) and randomly assign probabilities to all the primitives within each selected constraint.

Table II presents the results of tests conducted on randomly generated PIA networks. For each test we run the solving method on 10 instances of the same problem and we take the average running time in seconds. The generated problems are complete PIA networks (all constraints are different from the universal relation I). Thus, the number of constraints is equal to \( N(N-1)/2 \) where N is the number of variables. The second column corresponds to the time needed by the path consistency algorithm performed before the backtrack search. The third column corresponds to the time to find the first solution. The fourth, fifth and sixth columns correspond respectively to the time needed to find the most robust solution for problems where 10%, 20% and 30% of their constraints are uncertain.

As we can see from the test results, after finding the first solution, not much effort is needed to find the most robust one while there is actually a large number of probable worlds and their corresponding solutions especially in the case of large size problems. This is due to the branch and bound algorithm that prunes the search space significantly and also to the fact that the first solution obtained is of good quality which sets the lower bound of the algorithm to a good value.

**V. CONCLUSION**

In this paper we have proposed a new framework for representing and solving PIA networks. The uncertainty is represented here by the probability of the existence of each Allen primitive within its uncertain constraint. A PIA network P will thus involve a list of worlds and solving P consists of finding a solution for the most probable ones. This is an optimization problem that we tackle using a branch and bound algorithm based on temporal constraint propagation. Experimental study on randomly generated uncertain temporal networks demonstrates the efficiency of our solving method.

In the near future we intend to extend the uncertainty to other types of temporal constraints. More precisely, our first goal will be to handle the uncertainty at both the numeric and symbolic levels. We will then see how this can be done in a dynamic environment (when temporal constraints are added and removed during the resolution of the probabilistic temporal problem). Another perspective is to consider (instead
of a branch and bound based algorithm) approximation methods such as Stochastic Local Search (SLS)[17], Genetic Algorithms (GAs)[18] and Ant Colony Algorithms (ACAs)[19]. While these techniques do not always guarantee an optimal solution to the probabilistic temporal problem, they are very efficient in time (comparing to branch and bound) and can thus be useful if we want to trade the optimality of the solution for the time performance.

TABLE II
TIME PERFORMANCE OF THE SOLVING METHOD.

<table>
<thead>
<tr>
<th>N</th>
<th>PC</th>
<th>First Solution</th>
<th>10% uncertain</th>
<th>20% uncertain</th>
<th>30% uncertain</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.05</td>
<td>7.294</td>
<td>3.095</td>
<td>3.075</td>
<td>4.689</td>
</tr>
<tr>
<td>60</td>
<td>0.15</td>
<td>6.714</td>
<td>7.21</td>
<td>7.03</td>
<td>8.35</td>
</tr>
<tr>
<td>80</td>
<td>0.35</td>
<td>15</td>
<td>16.5</td>
<td>17.2</td>
<td>19</td>
</tr>
<tr>
<td>100</td>
<td>0.66</td>
<td>33</td>
<td>35</td>
<td>41</td>
<td>45</td>
</tr>
<tr>
<td>120</td>
<td>1.11</td>
<td>48</td>
<td>52</td>
<td>54</td>
<td>57</td>
</tr>
<tr>
<td>150</td>
<td>2.10</td>
<td>62</td>
<td>69</td>
<td>74</td>
<td>81</td>
</tr>
<tr>
<td>180</td>
<td>3.59</td>
<td>187</td>
<td>201</td>
<td>231</td>
<td>254</td>
</tr>
<tr>
<td>200</td>
<td>4.92</td>
<td>312</td>
<td>352</td>
<td>418</td>
<td>516</td>
</tr>
</tbody>
</table>

REFERENCES