Path Consistency on Triangulated Constraint Graphs*

Christian Bliek
ILOG
1681 Route des Dolines
06560 Valbonne, France
bliek@ilog.fr

Djamila Sam-Haroud
Artificial Intelligence Laboratory
Swiss Federal Institute of Technology
1015 Lausanne, Switzerland
haroud@lia.di.epfl.ch

Abstract

Among the local consistency techniques used in the resolution of constraint satisfaction problems (CSPs), *path consistency* (PC) has received a great deal of attention. A constraint graph G is PC if for any valuation of a pair of variables that satisfy the constraint in G between them, one can find values for the intermediate variables on any other path in G between those variables so that all the constraints along that path are satisfied. On complete graphs, Montanari showed that PC holds if and only if each path of length two is PC. By convention, it is therefore said that a CSP is PC if the completion of its constraint graph is PC. In this paper, we show that Montanari's theorem extends to triangulated graphs. One can therefore enforce PC on sparse graphs by triangulating instead of completing them. The advantage is that with triangulation much less universal constraints need to be added. We then compare the pruning capacity of the two approaches. We show that when the constraints are convex, the pruning capacity of PC on triangulated graphs and their completion are identical on the common edges. Furthermore, our experiments show that there is little difference for general non-convex problems.

1 Introduction

The constraint satisfaction paradigm allows for a natural formulation of a wide variety of practical problems. It consists of representing a problem as a set of variables taking their values in particular domains, subject to constraints which specify consistent value combinations. Solving a CSP amounts to assigning to the variables, values from their domains, so that all the constraints are satisfied. Backtrack search is the principal mechanism for solving a CSP. It is commonly combined with local consistency techniques to limit the combinatorial explosion. These techniques reduce the size of the search space by removing local inconsistencies.

*Authors are listed in alphabetical order.

This paper considers a particular form of local consistency called *path consistency* (PC). The work presented

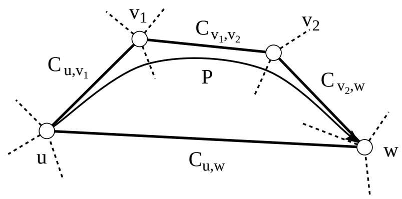

Figure 1: Path Consistency

considers binary CSPs and builds on their classical *constraint graph* representation, where the vertices represent the variables and the edges represent the constraints between the variables.

A path $P = \langle u, \ldots, v_i, \ldots, w \rangle$ in a constraint graph G is PC if for all pairs of values for (u, w) that satisfy the constraint $C_{u,w}$ in G one can find values for the intermediate variables v_i so that all the constraints $C_{u,v_1}, \ldots, C_{v_i,v_{i+1}}, \ldots, C_{v_k,w}$ in G along the path are satisfied (see figure 1). A constraint graph is PC iff all paths in the graph are PC [Mackworth, 1977]. In this paper we make the distinction between enforcing PC on CSPs and on constraint graphs. A CSP is PC if the completion of its constraint graph is PC. A CSP will be said to be *partially* PC (PPC) if its constraint graph is PC.

In practice, we know how to enforce PC on complete graphs thanks to the following theorem:

Theorem 1 *(Montanari, 1974) A network with a complete graph is PC iff every path of length two is PC.*

As a result, existing algorithms first complete sparse graphs by adding universal binary constraints, then enforce PC on each path of length two. The algorithms with the best time complexity [Mohr and Henderson, 1986; Han and Lee, 1988] run in time $O(n^3d^3)$. n is the number of variables and d the maximum domain size.

Despite its relatively high computational complexity, PC on complete graphs has been shown to be a central notion for certain classes of problems. In effect, it has been shown to be equivalent to global consistency for convex binary problems. This means that if a binary convex CSP is PC, its solutions can be derived backtrack-free. The PC property has also received particular attention in the area of temporal reasoning [Schwalb and Dechter, 1997] where lower forms of consistency prove to be of less interest.

In this paper we show that Montanari's theorem extends to triangulated constraint graphs. Triangulated constraint graphs can be made PC by ensuring that every path of length two is PC. In this case there is no need for additional constraints to be synthesized. This allows us to devise an algorithm for making a CSP with a triangulated constraint graph PPC in time $O(\delta e d^3)$. δ is the maximum degree of the graph and e is the number of edges in the graph. When the original constraint graph is not triangulated, we can triangulate it by adding universal edges. It is important to note that for sparse problems, the number of edges added by triangulation is much less than by completion.

Given an incomplete constraint graph, we then compare the pruning capacity of PC depending on whether it is enforced on a triangulation or a completion of the graph. We prove that for convex problems, the pruning capacity of the two is identical on the common edges. This means that, in this case, the extra edges synthesized for completion do not affect the labeling of the common edges. We also propose an algorithm for filling in these extra edges.

Finally, we present some experiments illustrating that significant gains in computational effort can be obtained using our algorithms. Furthermore it appears that there is little difference in the pruning capacity of PPC and PC for general non-convex CSPs with triangulated constraint graphs.

2 Background

In this paper, we consider binary CSPs (V, C, D) where V is the set $\{1, \ldots, i, \ldots, n\}$ of variables, D is the set $\{D_1, \ldots, D_i, \ldots, D_n\}$ of domains and variable i takes its value in domain D_i. The variables of V are subject to a set of constraints $C = \{C_{i,j} \mid C_{i,j}$ represent the legal value combinations from $D_i \times D_j\}$. We use the $(0,1)$ matrix representation of constraints proposed in [Montanari, 1974] and assume that $C_{i,j}$ is always the transposition of $C_{j,i}$. $C_{i,j} = D_i \times D_j$ is called a *universal* constraint. A constraint is *connected row-convex* (CRC) if after removing the empty rows from its matrix representation it is row-convex and connected, i.e. all the 1 entries in a row are consecutive and two successive rows either intersect or are consecutive [Deville *et al.*, 1997].

A CSP is *strongly* PC if in addition to PC it also is *arc consistent* (AC). A CSP is *globally* consistent if any partial instantiation of a subset of variables can be extended to a solution without backtracking.

Let us now recall the necessary background from graph theory. An undirected graph G is *triangulated* if every cycle of length strictly greater than 3 possesses a chord, that is, an edge joining two non-consecutive vertices of the cycle. For a graph $G = (V, E)$, with $|V| = n$, an ordering $[v_1, \ldots, v_i, \ldots, v_n]$ of V is a bijection of $\{1, \ldots, n\}$ onto V. For each v in V, the adjacency set $\mathrm{Adj}(v)$, is defined as $\{w \in V \mid (v, w) \in E\}$. A vertex v is *simplicial* if $\mathrm{Adj}(v)$ is complete. Every triangulated graph has a simplicial vertex. A triangulated graph remains triangulated after removing a simplicial vertex and its incident edges from the graph. The order in which simplicial vertices are successively removed is called a *perfect elimination order*. For a given perfect elimination order, we will use the notation $S_i = \{v_{n-i+1}, \ldots, v_n\}$, and G_i will denote the subgraph of G induced by S_i. $F_i = \{v_k \in \mathrm{Adj}(v_{n-i}) \mid v_{n-i} < v_k\}$ where $<$ is the precedence relation of the given order. Observe that since the elimination order is perfect, the subgraph of G induced by F_i is complete.

The material cited below is taken from[Kjærulff, 1990]. A perfect elimination order can be found in $O(n+e)$ time using the *maximum cardinality search* algorithm. A non-triangulated graph can always be transformed into a triangulated one by adding edges. Finding a *minimal* triangulation, where every edge is necessary for the graph to be triangulated can be done in $O(n(e+f))$ time, where f is the number of added edges. This bound is improved on average by a procedure called *recursive thinning* which we use in this work.

3 PC on Triangulated Constraint Graphs

In this section we extend theorem 1 to triangulated graphs. We show the following result:

Theorem 2 *A triangulated constraint graph G is PC iff every path of length 2 is PC.*

Proof: Since G is triangulated, we can find a perfect elimination order which defines S_i, G_i and F_i as discussed above. We demonstrate that G is PC by induction on i. Since every path of length 2 is PC, we know by construction that G_3 is PC. Assuming that G_i is PC, we set out to prove that G_{i+1} is PC. We do this by showing that any path P from u to w in G_{i+1} is PC.

If P is in G_i then P is PC by assumption. So we need to consider two cases. Either 1) as illustrated on the left in figure 2, v_{n-i} is an endpoint of P, e.g. $v_{n-i} = w$ or 2) P goes through v_{n-i} as shown on the right in figure 2.

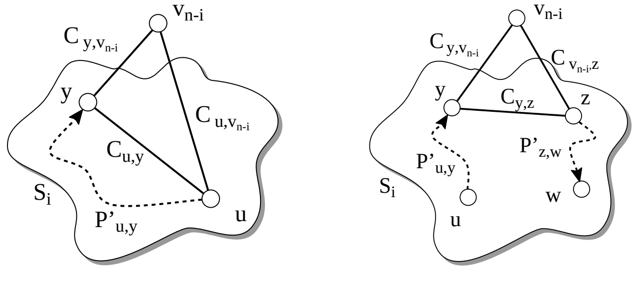

Figure 2: Two cases of inductive proof

Let us consider these two cases below:

1. This path is considered for PC only if there is a constraint $C_{u,v_{n-i}}$. Let us show that one can find values for the intermediate variables so that all the constraints along a path P are satisfied. Let y be the variable that precedes v_{n-i} in P. So in addition to the constraint $C_{u,v_{n-i}}$ we also have a constraint $C_{y,v_{n-i}}$. Since G is triangulated, the graph induced by F_i is complete. So there is a constraint $C_{u,y}$. Now in G every path of length 2 is PC, so for every pair of values that satisfy $C_{u,v_{n-i}}$ we can find a value for v that satisfies $C_{u,y}$ and $C_{y,v_{n-i}}$. The part $P'_{u,y}$ of the path P between u and y is in G_i and is therefore PC by assumption. So for the pair of values found for (u, y) we know that we can find values for the intermediate variables on this path $P'_{u,y}$. Hence we are able to find a set of values for all the intermediate variables between u and v_{n-i} that satisfy the constraints along P.

2. If P goes through v_{n-i} then let y and z be the variables that respectively precede and follow v_{n-i} in the path P. Note that both y and z are in S_i. Since there is a constraint $C_{y,v_{n-i}}$, a constraint $C_{v_{n-i},z}$ and that the graph induced by F_i is complete, we know that there is a constraint $C_{y,z}$. Now consider the path $P'_{u,y}; P'_{z,w}$ that goes directly from y to z without passing through v_{n-i}. $P'_{u,y}; P'_{z,w}$ is in G_i and therefore by assumption PC. This means that for any pair of values for (u, w) we can find values for the intermediate variables so that all the constraints along $P'_{u,y}; P'_{z,w}$ are satisfied. Since in G every path of length 2 is PC, we also know that for the pair of values found for (y, z) we can find a value for v_{n-i} that satisfies both $C_{y,v_{n-i}}$ and $C_{v_{n-i},z}$. By doing so we just have found a set values that satisfy all the constraints along the original path P. $\square$

4 From Triangulated to Completed Graphs

Given the result in the previous section, the question arises whether more pruning can be obtained by completing a triangulated graph. As demonstrated in section 6 this may indeed occasionally occur. However, in this section we show that for the class of convex problems, no additional pruning is obtained by completing the graph. The notion of convexity we refer to is a broad one. It includes the conventional definition of convexity in continuous domains, as well as its CRC extension to the discrete case. This extended convexity property is closed under composition and intersection of constraints.

To show our result on convex problems we need the following lemma.

Lemma 1 *If $G = (V, E)$ is an incomplete triangulated graph, then one can add a missing edge (u, w), with $u, w \in V$ so that*

1. *the graph $G' = (V, E \cup \{(u, w)\})$ is triangulated and*
2. *the graph induced by $X = \{x \mid (u, x), (x, w) \in E\}$ is complete.*

Proof: Since G is triangulated, it has a perfect elimination order defining S_i, G_i and F_i. Let i be the smallest index such that G_i is complete. Consider a variable $v_j \in S_i$ for which there is no edge (v_{n-i}, v_j) in G. By taking $(u, w) = (v_{n-i}, v_j)$ we now prove the two claims of the lemma.

1. Since G_i is complete, there is an edge between v_j and every variable in F_i. The graph induced by F_i in $G' = (V, E \cup \{(v_{n-i}, v_j)\})$ therefore remains complete. As a result, the considered elimination order is also perfect for G'. Since a graph with a perfect elimination order is triangulated, G' is triangulated.

2. Let us first show that $X = F_i$. Suppose it is not. In that case we necessarily have that a $y \in X$ that precedes v_{n-i} for which $(v_{n-i}, y), (y, v_j) \in E$. But then, since G is triangulated, there would also be an edge (v_{n-i}, v_j) which contradicts our assumption. Finally, since $X = F_i$ and G is triangulated, we know that the graph induced by X is complete. $\square$

This lemma is illustrated in figure 3. In this case

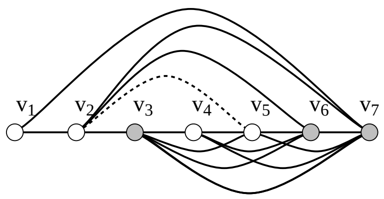

Figure 3: Completing Triangulated Graphs

$S_i = \{v_3, v_4, v_5, v_6, v_7\}$, $v_{n-i} = v_2$ and $F_i = \{v_3, v_6, v_7\}$. The variables of F_i are colored gray. The construction would for example add the dashed edge (v_2, v_5) which is currently missing.

Let us now turn to the main result of this section.

Theorem 3 *For a convex CSP with a triangulated constraint graph G, strong PC on G is equivalent to strong PC on the completion of G.*

By *equivalent* we mean that the relations computed for the constraints in G are identical.

Proof: Suppose we have a triangulated graph $G = (V, E)$ that is strongly PC. We will add to G the missing edges one by one until the graph is complete. To prove the theorem, we show that the relations of the constraints can be computed from the existing ones so that each intermediate graph, including the completed graph, is strongly PC.

To add the edges, we use the construction proposed for the proof of lemma 1. At all times during the constraint addition process, the graph therefore remains triangulated. After the addition of a single edge (v_{n-i}, v_j) to G, we obtain $G' = (V, E \cup \{(v_{n-i}, v_j)\})$. For this edge the new relation C_{v_{n-i},v_j} is computed as follows:

$$C_{v_{n-i},v_j} = \bigcap_{v_k \in F_i} C_{v_{n-i},v_k} \otimes C_{v_k,v_j} \tag{1}$$

where $\otimes$ is the composition operator. For example, in figure 3, C_{v_2,v_5} is obtained by intersecting the compositions obtained via the variables in $F_i = \{v_3, v_6, v_7\}$. C_{v_{n-i},v_j} is the universal relation when $F_i = \emptyset$.

If after making G strongly PC the relations in G are empty, the construction above would, as desired, compute the empty relations for the missing edges. In what follows we therefore assume that the relations in G after strong PC are not empty.

We now show that G' is PC. Since G' is triangulated, by theorem 2 it is sufficient to prove that every path of length 2 is PC. By assumption, paths of length 2 that do not go through v_{n-i} and v_j are PC. So let us consider paths of length 2 that go through v_{n-i} and v_j. By lemma 1 and the construction used in its proof, we know that the set of intermediate variables on the relevant paths is F_i and that F_i induces a complete subgraph of G. This situation is illustrated in figure 4, where the variables in F_i are colored gray. Note that by construction the graph A induced by $\{v_{n-i}\} \cup F_i$ and the graph B induced by $F_i \cup \{v_j\}$ are complete. We have to consider

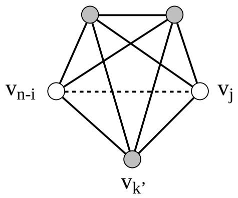

Figure 4: Added edges are PC

two cases[1]. With $v_{k'} \in F_i$, either 1) $P = \langle v_{n-i}, v_j, v_{k'} \rangle$ or 2) $P = \langle v_{n-i}, v_{k'}, v_j \rangle$. Let us consider each of these cases in turn.

1. If $P = \langle v_{n-i}, v_j, v_{k'} \rangle$ we need to prove that for every pair of values for $(v_{n-i}, v_{k'})$ that satisfies $C_{v_{n-i},v_{k'}}$ we can find a value for v_j so that C_{v_{n-i},v_j}, as defined by (1), and $C_{v_j,v_{k'}}$ are satisfied.

   A is complete, strongly path-consistent and convex, it is therefore also globally consistent [Sam-Haroud and Faltings, 1996]. So that for every pair of values that satisfy $C_{v_{n-i},v_{k'}}$ we can find values for all the variables in $F_i - \{v_{k'}\}$ so that all other edges in A are also satisfied (see figure 4). Similarly, since B is complete, strongly path-consistent and convex it is globally consistent. This means that for the above values of the variables in F_i, we can find a value for v_j so that all constraints in B are satisfied. For the considered values the constraint $C_{v_j,v_{k'}}$ is therefore satisfied. For any pair of values for $C_{v_{n-i},v_{k'}}$ we are hence able to find values for the variables in $(F_i - \{v_{k'}\}) \cup \{v_j\}$ so that all constraints in $A \cup B$ are satisfied. For the values for (v_{n-i}, v_j), the relations C_{v_{n-i},v_k} and C_{v_k,v_j} participating in (1) are therefore satisfied by the values for the variables in F_i. The values for (v_{n-i}, v_j) therefore satisfy C_{v_{n-i},v_j}.

2. If $P = \langle v_{n-i}, v_{k'}, v_j \rangle$ the relation C_{v_{n-i},v_j} of the new edge ensures by definition that for every pair of values in C_{v_{n-i},v_j} we can find a value for $v_{k'}$ so that the $C_{v_{n-i},v_{k'}}$ and $C_{v_{k'},v_j}$ are verified.

We now show that G' is AC. When $|F_i| \geq 1$, since G' is PC we know that for every value pair in $C_{v_{n-i},v_{k'}}$ or in $C_{v_{k'},v_j}$ we can find a value pair that satisfies C_{v_{n-i},v_j}. Since G is AC this means that for any support respectively on $C_{v_{n-i},v_{k'}}$ or on $C_{v_{k'},v_j}$ we are able to find a support on C_{v_{n-i},v_j}. As a consequence G' is not only PC but also AC. When $|F_i| = 0$, $C_{v_{n-i},v_j} = D_{v_{n-i}} \times D_{v_j}$. Since G is AC, G' will therefore be AC as well. $\square$

Corollary 1 is a direct consequence of theorem 3.

Corollary 1 *For convex problems, insolubility is detected using PC by graph completion iff it is detected using PC by graph triangulation.*

5 Algorithms

By theorem 2 we know that triangulated graphs can be made PC by enforcing that every path of length two is PC. For problems with a triangulated constraint graph we can therefore make the CSP PPC by a simple modification of existing PC algorithms[2]. The resulting algorithm is Algorithm 1. The procedure *Related-Triplets*(q) returns all those triplets in which q participates and that correspond to actual triangles in G. The difference with a classical PC algorithm, referred to as PC in the rest of the paper, is that PC revises all possible triplets, not only those corresponding to triangles in G. To determine the complexity of this algorithm for discrete problems, consider the number of revisions that can be made based on each edge (v_i, v_j). Revisions based on C_{v_i,v_j} will be made only when a pair of values is removed. If d is the maximum domain size, then one can remove at most d^2 pairs from any relation. Each removal

algorithm PPC
  $Q \leftarrow E$
  Until Q *is empty* **do**
    $q \leftarrow$ Dequeue(Q)
    for $(v_i, v_k, v_j) \in$ *Related-Triplets*(q) **do**
      $C_{v_i,v_j} \leftarrow C_{v_i,v_j} \cap (C_{v_i,v_k} \otimes C_{v_k,v_j})$
      if C_{v_i,v_j} *has changed* **then**
        Enqueue((v_i, v_j), Q)
      end
    end
  end
end.

Algorithm 1: PPC

[1] The reasoning is the same for the symmetrical cases.

[2] PPC is close to Schwalb and Dechter's PLPC algorithm from which it borrows the name. PLPC (Partial Loose PC) enforces a partial form of path consistency on disjunctive temporal CSPs. It only considers the paths of length two with at least two non-universal constraints.

prompts revisions only of the 2 neighboring edges in each triangle. If δ_{v_k} is the degree of variable v_k, a modification of C_{v_i,v_j} will prompt at most $2\min\{\delta_{v_i}-1,\delta_{v_j}-1\}$ revisions. Summing over all edges we find that at most

$$\sum_{(v_i,v_j)\in E} 2\min\{\delta_{v_i}-1,\delta_{v_j}-1\}d^2 = O(\delta e d^2)$$

revisions will be performed, where δ is the maximum degree. This should be compared to the $O(n^3d^2)$ revisions performed by the classical PC algorithm. Using the results presented in [Chiba and Nishizeki, 1985], one may use the arboricity α of G instead of δ, resulting in a number of revisions $O(\alpha e d^2)$. The same reference also presents upper bounds on α both for general graphs and for specific types of graphs.

For the experiments below we report the number of revisions since this measure is independent of the specific techniques one might use for updating the relations. In practice one can for example use the techniques based on the principle of minimal support used in PC-6 [Chmeiss, 1996]. In this case at most $O(ed^2)$ value pairs may be deleted in the relations. Per value pair at most $O(\delta d)$ supports may be visited leading to a time complexity of $O(\delta e d^3)$. However, per value pair only $O(\delta)$ support information concerning the smallest supporting element is stored, resulting in a $O(\delta e d^2)$ space complexity. In case of CRC constraints one can use the techniques described in [Deville *et al.*, 1997] to obtain a time complexity of $O(\delta e d^2)$ and a space complexity of $O(\delta e d)$.

The PPC algorithm makes CSPs with triangulated constraint graphs PPC. CSPs whose constraint graph is not triangulated can be made PPC by triangulating the graph with universal constraints before running PPC. For convex problems PPC is equivalent to PC. If desired, the relations of the missing edges can be computed as proposed in the proof of theorem 3. This fill algorithm is shown in Algorithm 2 below.

algorithm Fill
- **for** $i \leftarrow 1$ ***to*** n **do**
  - **Until** G_i *is complete* **do**
    - let (v_{n-i}, v_j) be a missing edge in G_i
    - $C_{v_{n-i},v_j} \leftarrow \bigcap_{v_k\in F_i} C_{v_{n-i},v_k} \otimes C_{v_k,v_j}$
  - **end**
- **end**

end.

Algorithm 2: Fill

The variables are assumed to be ordered according to a perfect elimination order. Observe that it is not required to update the sets F_i when G changes, since the order in which the edges are added to G_i does not matter.

Algorithm 2 computes at most $n(n-1)/2$ relations. Since each relation is computed by intersecting at most 2δ compositions, the number of revisions Algorithm 2 needs to perform is therefore $O(\delta n^2)$. For CRC constraints one can, as before, use the techniques described in [Deville *et al.*, 1997] to perform the actual computations. In this case the missing relations can be filled in in time $O(\delta n^2 d^2)$.

6 Experiments

In this section, we report some preliminary experiments that compare the number of revision steps carried out by PC and PPC for three types of randomly generated problems. The convex case is illustrated by tests on linear continuous and CRC problems, the non-convex one by tests on randomly generated discrete problems. For the linear problems, the constraints generated are inequalities. They are discretized similarly to what is described in the work of [Sam-Haroud and Faltings, 1996] and represented by $(0,1)$ matrices. The constraint graph of each generated instance is triangulated before running PPC. The domain size is 8 for all the types of problems. Each test is averaged over 25 instances.

Table 1 shows the comparison for different sizes of constraint graphs and a fixed density p. The density chosen ($p = 0.1$) corresponds to sparse graphs and illustrates the most favorable case for PPC. Table 2 compares PC and PPC for problems of fixed size ($n = 20$) and different densities. Since PPC is of interest on sparse CSPs, we report the experiments on sparse graphs ($p \leq 0.5$). For higher densities, the number of revisions of PPC approaches that of PC. Note that PPC and PC are identical for complete graphs.

By theorem 3 for convex problems the relations computed by PPC and PC on the common edges are identical. For non-convex problems, we also compare the pruning capacity of PPC to the one of PC. ρ is the ratio between the number of tuples removed by PPC over the number of tuples removed by PC on the common edges. For the tests conducted on random problems, insolubility detected by PC was also detected by PPC.

The tests reported for non-convex problems are generated in the phase transition [Grant and Smith, 1996] as this seems to best illustrate the difference of behavior between PPC and PC. Indeed, several hundreds of tests run out of the phase transition showed no difference of pruning between PC and PPC on the common edges. Note that the randomly generated linear and CRC problem instances do not necessarily fall in the phase transition.

It is worth mentioning that despite the worst case complexity of $O(n(e+f))$ for triangulation algorithms, the effective time devoted to triangulation is negligible compared to the running time of PPC.

More experiments clearly need to be conducted for better stating the effectiveness of PPC. The preliminary results we report are however encouraging enough to warrant PPC being investigated as alternative to PC for sparse problems.

7 Conclusion

Path consistency is an important notion in constraint satisfaction. A new algorithm, called PPC, is proposed that makes triangulated constraint graphs PC. PC can

Linear

n	p	PC	PPC
10	0.1	1,944	211
15	0.1	10,293	334
20	0.1	33,220	550
25	0.1	85,463	644
30	0.1	183,253	1,641
35	0.1	356,825	4,532
40	0.1	619,999	9,666

CRC

n	p	PC	PPC
10	0.1	1,827	222
15	0.1	9,117	374
20	0.1	32,213	559
25	0.1	84,796	950
30	0.1	185,807	1,825
35	0.1	361,000	5,160
40	0.1	624,314	14,621

Random

n	p	PC	PPC	ρ (%)
10	0.1	1,971	339	99.75
15	0.1	13,870	407	99.51
20	0.1	32,366	749	99.65
25	0.1	97,180	1,388	99.84
30	0.1	143,116	5,441	99.80
35	0.1	348,399	5,477	100.0
40	0.1	665,871	33,623	99.92

Table 1: Revisions performed by PC and PPC on sparse graphs for different problem sizes

Linear

n	p	PC	PPC
20	0.1	33,220	550
20	0.2	37,440	1,660
20	0.3	38,187	5,329
20	0.4	38,719	8,907
20	0.5	39,054	13,736

CRC

n	p	PC	PPC
20	0.1	32,213	559
20	0.2	37,593	2,257
20	0.3	38,393	5,692
20	0.4	38,811	8,919
20	0.5	39,043	13,339

Random

n	p	PC	PPC	ρ (%)
20	0.1	32,366	749	99.65
20	0.2	35,017	3,263	99.98
20	0.3	55,330	13,982	100.0
20	0.4	112,986	45,979	100.0
20	0.5	223,459	148,330	100.0

Table 2: Revisions performed by PC and PPC on given problem size for different densities

thus be enforced on incomplete constraint graphs by triangulating instead of completing them. When the problem is sparse, this spares a significant amount of work compared to the classical PC algorithm. We have also shown that for convex CSPs with triangulated constraint graphs, the PPC and PC algorithms will compute the same labeling on the common edges.

Excessive memory requirements of existing PC algorithms limit their applicability [Chmeiss and Jégou, 1996]. Since on sparse graphs PPC has a lower space complexity than PC, PPC might prove to be a viable alternative when memory is a limiting factor.

For non-convex problems, PPC exhibits a good pruning capacity compared to PC and can be computed much more efficiently than PC on sparse graphs. We therefore expect that it might be beneficial to interleave it with backtrack algorithms to search for solutions. This will be a topic of future research.

8 Acknowledgments

We would like to thank Jean-Charles Régin for pointing out the results on arboricity. Most of this work was performed at the Artificial Intelligence Laboratory of the Swiss Federal Institute of Technology in Lausanne where Christian Bliek was sponsored by the Swiss National Science Foundation under project number 2000-52363.97 through the ERCIM Fellowship Program.

References

[Chiba and Nishizeki, 1985] N. Chiba and T. Nishizeki. Arboricity and subgraph listing algorithms. *SIAM Journal on Computing*, 14, 1985.

[Chmeiss and Jégou, 1996] A. Chmeiss and P. Jégou. Path-consistency: When space misses time. In *AAAI-96*, pages 196–201, Portland, Oregon, 1996.

[Chmeiss, 1996] A. Chmeiss. Sur la consistance de chemin et ses formes partielles. In *Actes du Congrès RFIA-96*, pages 212–219, Rennes, France, 1996.

[Deville *et al.*, 1997] Y. Deville, O. Barette, and P. Van Hentenryck. Constraint satisfaction over connected row convex constraints. In *IJCAI-97*, pages 405–410, Nagoya, Japan, 1997.

[Grant and Smith, 1996] S. A. Grant and B. Smith. The arc and path consistency phase transitions. In *CP-96*, pages 541–542, 1996.

[Han and Lee, 1988] C. Han and C. Lee. Comments on Mohr and Henderson's path consistency algorithm. *Artificial Intelligence*, 36, 1988.

[Kjærulff, 1990] U. Kjærulff. Triangulation of graphs – algorithms giving small total state space. Research Report R-90-09, Aalborg University, Denmark, 1990.

[Mackworth, 1977] A.K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8:99–118, 1977.

[Mohr and Henderson, 1986] R. Mohr and T.C. Henderson. Arc and path consistency revisted. *Artificial Intelligence*, 28:225–233, 1986.

[Montanari, 1974] Ugo Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Information Science*, 7:95–132, 1974.

[Sam-Haroud and Faltings, 1996] D. Sam-Haroud and B.V. Faltings. Consistency techniques for continuous constraints. *Constraints*, 1:85–118, 1996.

[Schwalb and Dechter, 1997] E. Schwalb and R. Dechter. Processing disjunctions in temporal constraint networks. *Artificial Intelligence*, 93:29–61, 1997.