ROUNDTRIP PUZZLES

Extended Introduction with solving strategies

by Glenn A. Iba

Introduction & Background

RoundTrip puzzles first appeared in Dell Magazines in the early 1990's. They were invented by a puzzler named Stitch (that's his nom in NPL, the National Puzzlers' League). His puzzles were presented as an 8x8 square grid of 64 dots with just enough links provided to make the solution unique. Links can only connect neighboring dots (horizontally or vertically, but not diagonally). The puzzle objective is always to form a closed path (one that returns to its starting point) that connects all the dots in a single large loop that visits each dot exactly once. The solution path is formed by drawing in additional links connecting neighboring dots. It is possible to solve RoundTrip puzzles using logical reasoning. Random guessing is not likely to succeed with any but the simplest puzzles.

I enjoyed solving Stitch's RoundTrip puzzles so much that I decided to write a computer program to generate more of them. I started by creating a program to solve the puzzles, and then used that to generate uniquely solvable puzzles. I also extended the program to generate puzzles using a triangular grid of dots (in which case interior dots have 6 neighbors compared to the 4 neighbors of square grids). I laid out the dots in a hexagon-shaped grid. I submitted puzzles of both shapes (square and hex) and various sized grids to Dell Puzzle Magazines. The first of my puzzles appeared in Dell Champion Variety Puzzles November 1992, and have appeared in various Dell Magazines since then. Some of my puzzles have also appeared (under the name "GrandTour") in a Page-a-Day Calendar edited by Scott Kim for Workman Publishing. Additionally, I have posted Grand Tour puzzles on my web site:

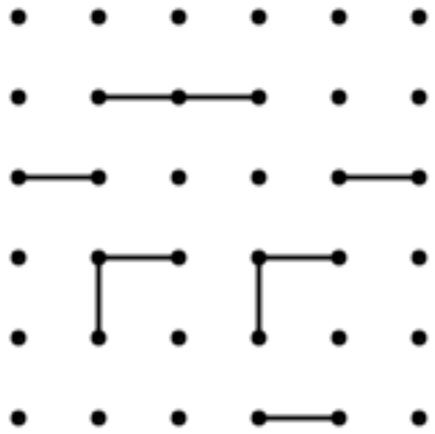
http://glenniba.com/

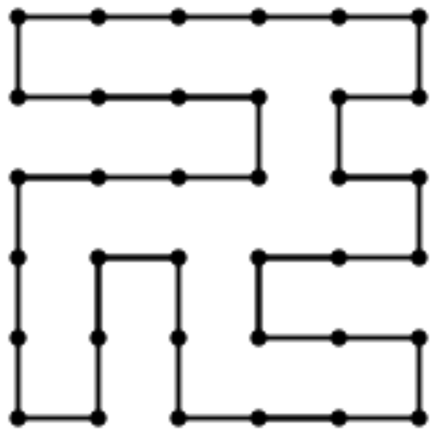
These puzzles can be solved interactively in a GrandTour Java Applet I wrote.

This book contains many examples of both square and hex RoundTrip puzzles. They range in difficulty from quite easy to very challenging. In a following section I include discussion of various solving techniques, both elementary and advanced. I hope you will have as much fun with these puzzles as I have.

The Elements of RoundTrip Puzzles

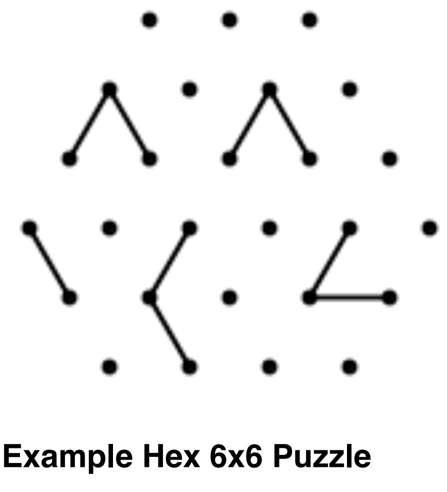
Here is a simple example of a square RoundTrip puzzle and its solution:

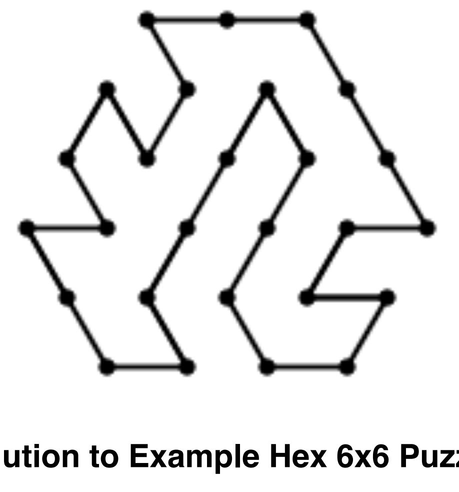

Example Square 6x6 Puzzle


Solution to Example Square 6x6 Puzzle

A RoundTrip puzzle consists of a **grid** of **dots** and a number of **required links** connecting neighboring dots. A solution is a single looping **path** connecting all of the dots, and closing on itself so that there are no endpoints. The links in the path must include all the required links given in the initial puzzle presentation, and each additional link must connect 2 neighboring dots. In this square 6x6 example, there are 36 dots and 9 required links given initially. Observe that the solution has 36 links including those given, and that every dot has exactly 2 links connecting to it. In this book you will find Square puzzles of sizes: 6x6, 8x8, 10x10, and 12x12.

This book also contains Hex RoundTrip puzzles of various sizes: 5x5, 6x6, 7x7, 8x8, 9x9, 10x10, and 11x11. Here is an example of a 6x6 Hex puzzle and its solution:

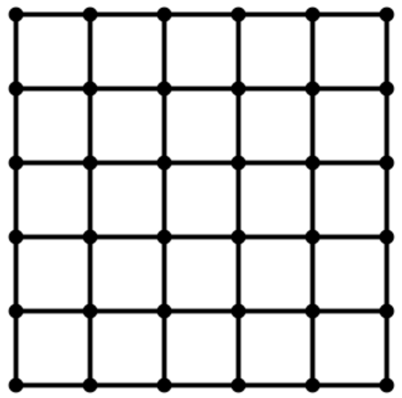

Example Hex 6x6 Puzzle


Solution to Example Hex 6x6 Puzzle

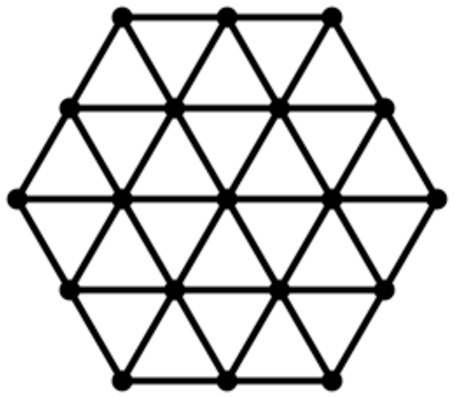
A Hex puzzle has its dots laid out in a grid shaped like a hexagon. In this Hex example there are 27 dots and 9 required links given. The solution consists of a total of 27 links. It will always be true that the number of links in a solution is the same as the number of dots in the puzzle grid. This is because every dot must be traversed exactly once.

Neighboring Dots

Links can only connect a dot with a **neighbor** dot. Informally, two distinct dots are **neighbors** if they are as close as possible in the grid. Here are sample square and hex grids with all the possible neighbors drawn in:

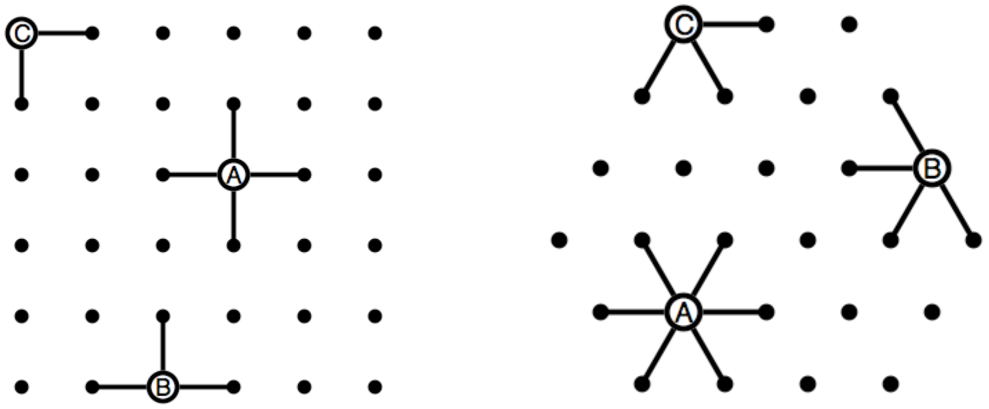


(Fig. N1) Square 6x6 with all neighbors



(Fig. N2) Hex 5x5 with all neighbors

There are 2 types of grid layout used in the RoundTrip puzzles of this book. They are illustrated in figures N1 and N2. The two types of grid are called Square and Hex respectively, due to the shape of the grid outline. Note that when all neighboring links are drawn, the square grid consists of square spaces, and in the hex grid consists of triangles. The following figures (N3 and N4) illustrate the fact that dots can have different numbers of neighbors depending on their position in the grid:



(Fig. N3) Neighbors of different Dot types: Interior (A), Border (B), and Corner (C)

Note that dots in the square grid can have 2, 3, or 4 neighbors. Diagonal links are not allowed. Interior dots will have 4 neighbors, dots at the edge or border of the grid have

3 neighbors, and corner dots have only 2 neighbors. In the hex grid, a dot may have 3, 4, or 6 neighbors. Interior dots have the full 6 neighbors, while border dots have 4, and the six corner dots have only 3 neighbors.

ADDITIONAL CONCEPTS:

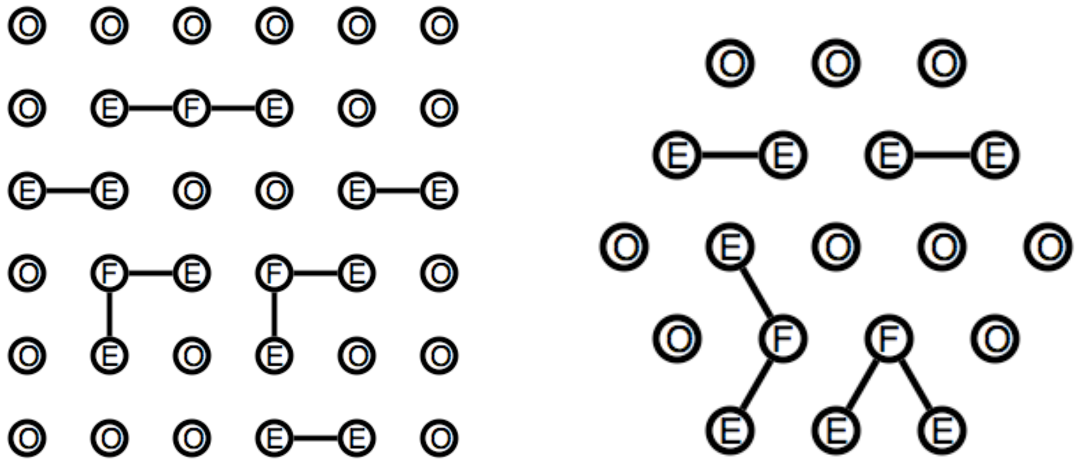
Dot Availability

In the initial puzzle and at every stage of solving, dots may be of 3 types, depending on how many links connect to the dot:

1. A dot with 0 links is called **open**
2. A dot with 1 link is an **endpoint**
3. A dot with 2 links is **filled**

We say that both Open Dots and Endpoints are **available**, because they can receive additional links. Filled Dots are **unavailable** since they are already "used up", and cannot accept additional links (since in a valid solution no dot can have more than 2 links). Note that Open Dots need 2 additional links, and Endpoints need just 1 additional link.

In the example puzzle grids of Figure N4, all the dots are labelled according to whether they are of type Open (O), Endpoint (E), or Filled (F).



(Figure N4) Availability of Dots: Open (O), Endpoint (E), or Filled (F)

Chains and Loops

A **chain** is a connected sequence of links connecting 2 endpoints. Except for the endpoints, each dot connected by the chain must have exactly 2 links. Every chain will have exactly 2 endpoints. Chains are illustrated in Figure N-chains.

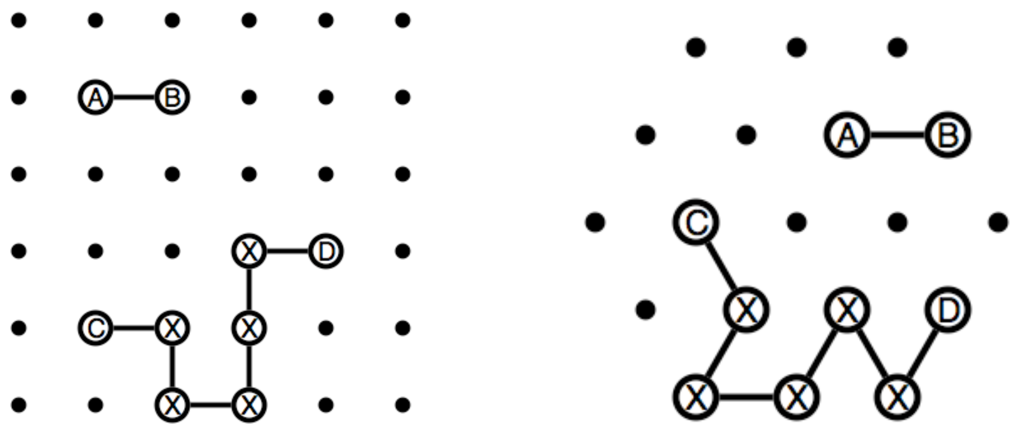


Figure N-chains: Examples of chains in both square and hex grids

The simplest example of a chain is a single link, such as A-B in the grids of figure N-chains. Dots A and B are the 2 endpoints. Longer chains are illustrated by the sequence of links connecting C and D and the dots labelled X. Here C and D are the endpoints of the chain, and all the dots labelled X form the **interior** of the chain. Note that each of the X dots is filled since they each have 2 links, and are therefore unavailable.

Chains can be extended by adding links to their endpoints. If a link is added from an endpoint to an open dot, then this extends the chain and the open dot now becomes a new endpoint of the chain. If a link connects 2 endpoints of distinct chains, this merges the 2 chains into a single longer chain.

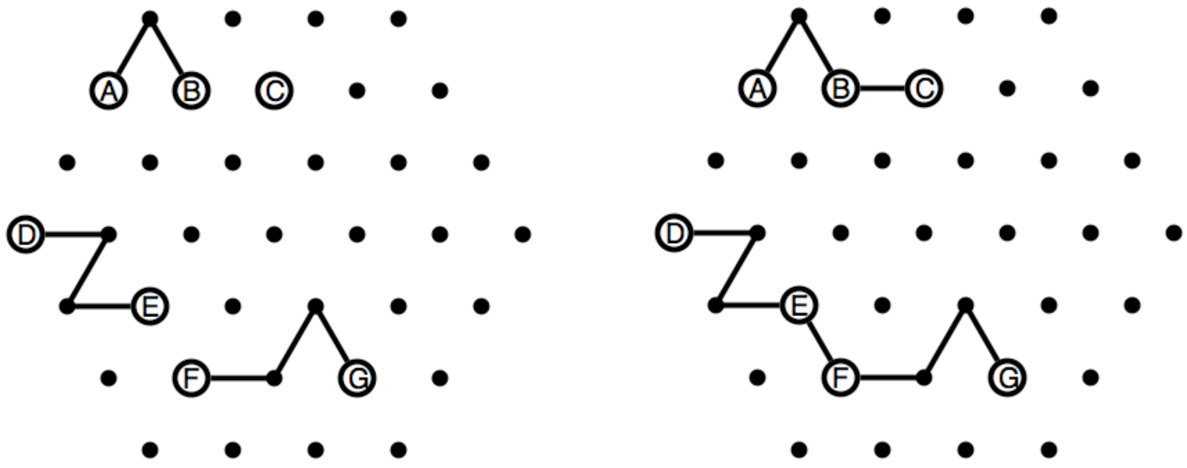
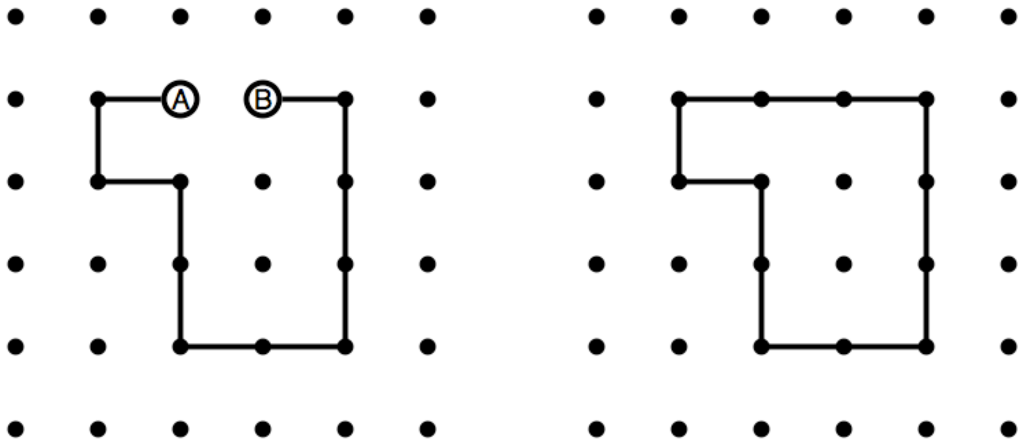


Figure Merging Chains: Before Merge at Left, After Merge on Right

These two types of chain merge are illustrated in Figure Merging Chains, where the situation before merge is shown at left, and the right shows the result of the merge. The 2-link chain A-B is extended by adding link B-C connecting endpoint B to the open dot C.   This creates the new 3-link chain A-C. Also, the 3-link chain D-E is merged with the 3-link chain F-G by adding the link E-F connecting 2 endpoints E and F. The result is one 7-link chain D-G

A **loop** is formed when a link is added connecting 2 neighboring dots which happen to be the 2 endpoints of a single chain. A loop has no endpoints, and every dot connected in the loop must have exactly 2 links. If the loop connects all the dots of the grid, then (congratulations!) you have solved the puzzle. If the loop does not include all of the dots, then this is a **short loop.** Short loops should be avoided, since they can never be part of a solution.



(Figure Loop Creation) A loop is created by connecting endpoints of Chain A-B.

Figure Loop Creation illustrates the formation of a Short Loop. The endpoints of Chain A-B are neighbors, so they could be connected with a new link. When link A-B is added the result is a loop. Notice that there are now no endpoints. This is a short loop because it does not include all the dots in the grid.

SOLVING STRATEGIES

RoundTrip puzzles can be solved by logical reasoning. Random guessing is extremely unlikely to yield a solution. The approach outlined here is to **only add forced links**, that is, links which must necessarily be part of the solution. Following this strategy implies that you will never need to erase any links you have drawn.

This explains why I have stated that dots can only have 2 links (at most). In a solution loop, all dots must have exactly 2 links, no more and no less. If 3 or more links were drawn to the same dot at some point while solving, then it would be necessary to erase all but 2 links in order to obtain a solution. Following the “add only” approach means never having to erase (unless you make a mistake).

In addition to identifying **forced links,** it is helpful to identify **forbidden links**, links which can be ruled out because they cannot ever become part of a solution (using the “add only” strategy). A simple example of a forbidden link would be any link to a dot which is already filled. Adding such a link would create 3 links at the dot (recall that a filled dot has 2 links by definition), which cannot lead to a solution by the “add only” strategy. It can be helpful, if you like, to mark forbidden links by placing a small **X** between the dots where a link cannot be placed.

Simple Forced Links

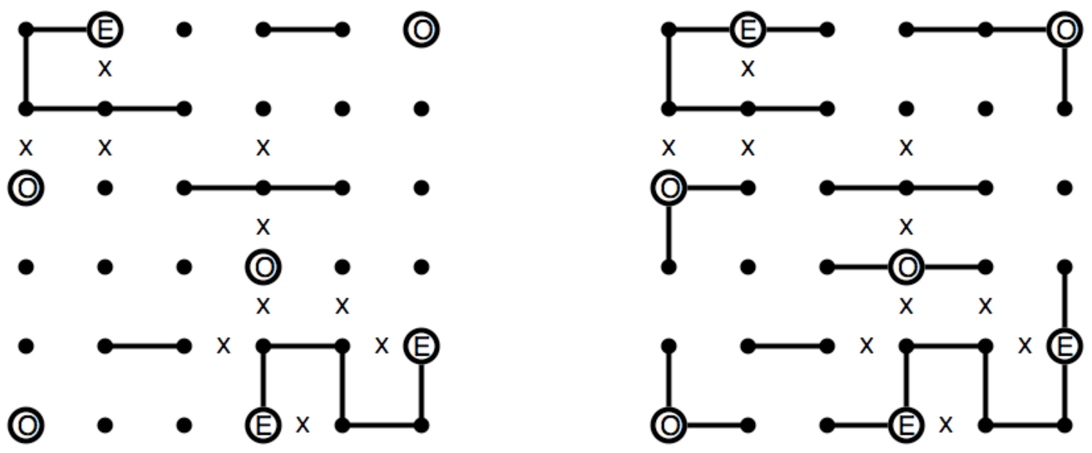
There are two elementary examples of forced links, one for endpoint dots, the other for open dots. You will be using these all the time, so it is worth understanding them and getting lots of practice finding them.

a. An ENDPOINT with only 1 available neighbor

Since an endpoint needs 1 more link as part of a solution, if it has only 1 neighbor available to connect with, then that connection is necessary to form a solution. That link is forced and can be added to your partial solution.

b. An OPEN dot with exactly 2 available neighbors

Since an open dot needs 2 more links, if there are exactly 2 available neighbors for the dot, then there must be links to both of these neighbors. You have found 2 forced links, which can be safely added.



Dots with forced links **After adding forced links**

Figure Simple-Forced-Links

In Figure Simple-Forced-Links the board is analyzed by first marking all forbidden links with an X. Then we look for endpoints with exactly 1 available neighbor, and open dots with exactly 2 available neighbors. There are 3 such endpoints (labelled **E** in the figure), and 4 such open dots (labelled **O**). The forced links have been added in the right hand side of the figure.

Avoid Impossible / Forbidden Patterns

In this section we identify three forbidden patterns, and look at how they can be used to identify forbidden links, which may lead to additional forced links. A **forbidden pattern** is some pattern of dots and links which makes a puzzle impossible to solve using the strategy of only adding links. Any forbidden patterns must be avoided, and any link that would create a forbidden pattern is therefore a forbidden link, and can be marked off with an **X**, if desired.

a. **3 or more links at 1 dot**

We have already encountered this rule. It just restates the rule that a dot cannot link to a filled neighbor. Any link to a filled dot is thus a forbidden link and can be ruled out.

b. Isolated Dots

An **isolated dot** is one which doesn't have enough available neighbors. Recall that in a solution, every dot must have exactly 2 links to neighboring dots. If an endpoint has no available neighbors (because they are already filled), then that endpoint is isolated. It can never reach its desired 2 links by adding links. Note that adding a link to such an endpoint would create a forbidden pattern of type **a** (3 links) at the neighboring dot. An open dot is isolated if it has fewer than 2 available neighbors. An open dot needs to receive 2 links in a solution, but if there are only 0 or 1 neighbors that are not filled, then adding any 2 links will have to cause some filled neighbor to have 3 links (forbidden by a). No solution is possible if there is an isolated dot of either type, so isolated dots are forbidden patterns. Since you can never add a link which creates an isolated dot, any such link is a forbidden link.

c. Short Loops

Short loops (loops not containing all the dots in a grid) are forbidden patterns, since they can never be extended to a solution where all the dots are connected in a single loop, as required.. Thus short loops must be avoided, and any link which would create a short loop can be ruled out as a forbidden link.

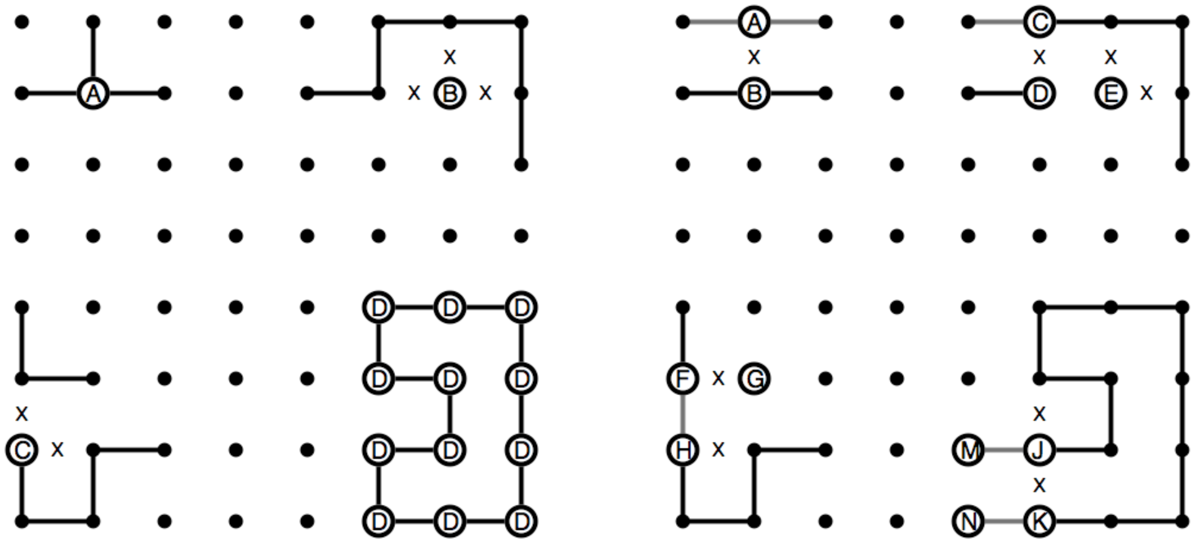


Figure: Simple Forbidden Patterns (left), Forbidden and Forced Links (right)

These 3 types of simple forbidden patterns are illustrated in the left part of Figure Forbidden Patterns. Dot A has 3 links, which is disallowed. Dot B is an isolated open dot. It has only 1 available neighbor. Dot C is an isolated endpoint, having no avaiilable neighbors. The dots labelled D form a short loop, the third type of simple forbidden pattern.

In the right half of the figure, we examine how avoiding forbidden patterns can identify forbidden links and lead to additional forced links. Link A-B would create the forbidden pattern of 3 links at B, so link A-B is forbidden. Having marked link A-B with an X, we now notice that dot A (an open dot) has only 2 available neighbors, so the gray links extending left and right from A are forced links. Link C-D would make dot E an isolated dot, which is forbidden, so link C-D can be ruled out. This in turn forces the gray link extending left from the endpoint C. Adding link F-G would make dot H an isolated endpoint, so link F-G is forbidden the only remaining available neighbor of F is now H, so link F-H is forced.

LookAhead

In the last section we discovered forbidden links by seeing that they immediately created a forbidden pattern. Sometimes we can identify a link as forbidden by looking ahead several steps. If the candidate link forces other links, which in turn force additional links, finally resulting in a forbidden pattern, then the candidate link can be ruled out as forbidden itself. The key point is that there is a sequence of forced links leading from the candidate link to a forbidden pattern.

As you become skilled at spotting forced links, forbidden links, and forbidden patterns, you should be able to look-ahead several steps without actually drawing the links. For many step lookahead (a strategy of last resort!) it may be necessary to draw in the "hypothetical" links, which should be drawn in a different color, or identified in some way as hypothetical so they can be erased later if need be.

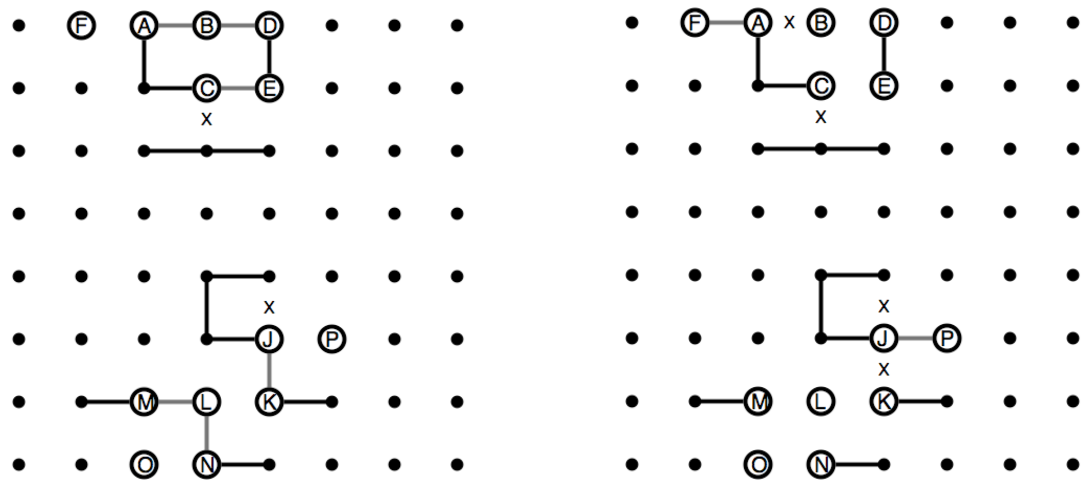


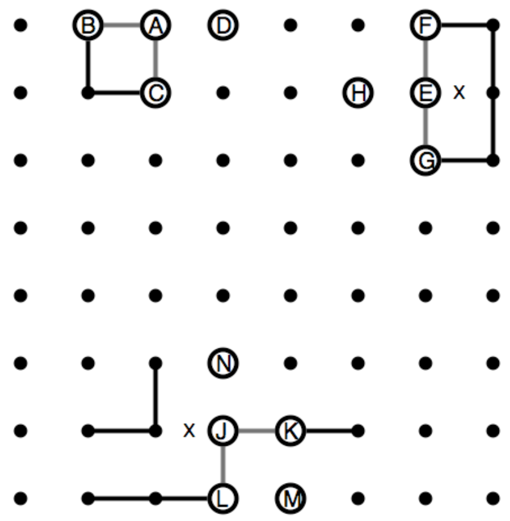
Figure: Hypothetical Lookahead (left), New forbidden and forced links (right)

Lookahead is illustrated in Figure Hypothetical Lookahead. At the top of the left diagram, we analyze candidate link A-B. Adding A-B would make link B-C forbidden (creates a short loop), so this forces B to link to D (only remainaing available neighbor) and C to link to E as well. This creates a short loop of 6 links which is forbidden. Since adding A-B leads by forces to a forbidden pattern, we conclude that A-B is a forbidden link, and mark it with an X (see right side of diagram). Now link A-F is forced since F is the only remaining available neighbor. [Note that B is no longer an available neighbor to A since the link A-B is forbidden]

In the bottom half of the figure, we consider the consequences of adding link J-K. If J-K were to be added, then L becomes an open dot with exactly 2 available neighbors M and N. Therefore the links L-M and L-N would be forced. After these links are added, we can see that open dot O is now isolated (it has only 1 available neighbor remaining). This is a forbidden pattern, so the original candidate link can be ruled out, that is, J-K is identified as a forbidden link. We mark J-K with an X in the right hand part of the diagram, and then observe that link J-P is now forced, since P is the only remaining available neighbor of J.

3-Way Force

The 3-way force is a somewhat subtle, but very useful pattern to recognize. It occurs when some open dot has exactly 3 available neighbors (and therefore 3 possible links to that dot), but you can rule out some **pair** of the 3 links, that is, the pair of links **cannot both be taken together** because they would lead (directly or by lookahead) to a forbidden pattern. We could say that such a pair of links is a **forbidden pair.** In such a case, the 3rd link is forced (that is, the link not in the forbidden pair is required). One of the links in the pair will have to be used eventually, but that is OK since only the **pair** was forbidden.



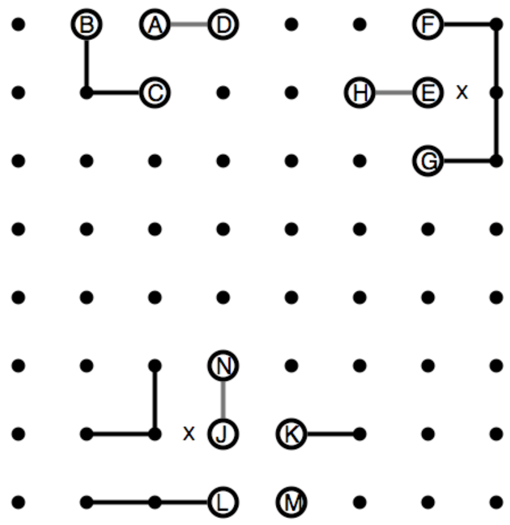
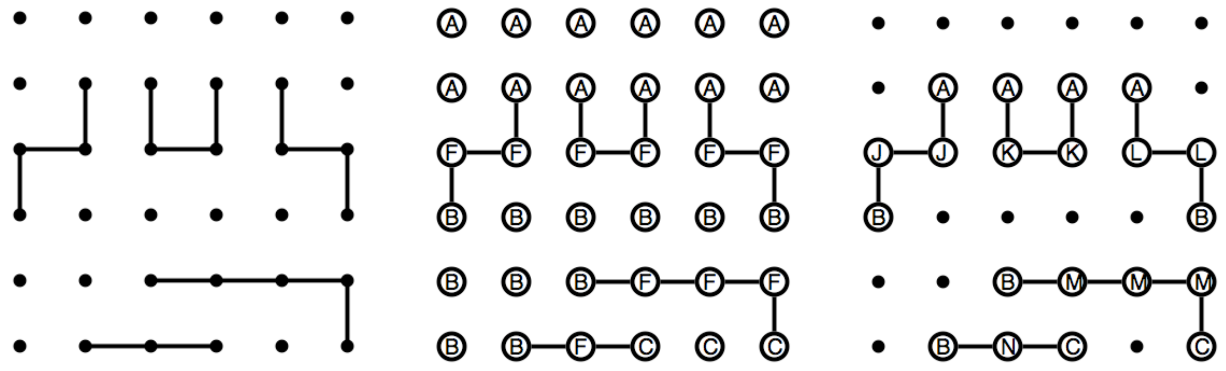


Figure 3-Way: Forbidden pairs (left), Resulting Forced links (right)

Several 3-Way forces are illustrated in Figure 3-Way. The left diagram shows the forbidden pairs in Gray. The right diagram shows the links that are forced by the 3-way reasoning. On the left, Dot A has exactly 3 available neighbors. If A were linked to both B and C, this would form a forbidden short loop, so the links A-B and A-C constitute a forbidden pair. Therefore the link A-D is forced as shown in gray on the right. Similarly, at Dot E (which has 3 available neighbors F, G, and H) the pair of links E-F and E-G are a forbidden pair since together they would create a short loop. We can conclude that the 3rd link E-H is thus forced (as shown in gray on the right). Finally, Dot J (left) has 3 available neighbors K, L, and N. The pair of links J-K and J-L, if taken together, would isolate Dot M. Therefore the link J-N is forced as shown on the right

Region Analysis

This section introduces an advanced, non-local, set of reasoning strategies for solving RoundTrip puzzles. These methods revolve around the analysis of **regions,** which are areas of the grid separated from other areas by **walls** of filled dots. Figure Regions-Analysis illustrates this concept. We say 2 unfilled dots are in the same region if there is some sequence of neighboring dots leading from one to the other without crossing any filled dots. Two dots are said to be in different regions if the only way to get from one to the other is by crossing over 1 or more filled dots. In Figure Region-Analysis, part (a) shows the grid and the unlabeled links, part (b) shows how the filled dots (F) divide the un-filled dots into 3 regions: A, B, and C.



(a) Unlabeled Links **(b) Regions** **(c) Chain types**

Figure: Region Analysis

We now look at the chains in the grid and how they relate to the regions. This is shown in part (c) of the figure. Chains can be of 2 types with respect to regions. We say a chain is an **internal chain** if both its endpoints lie in the same region. Chain K in the figure part c, is an internal chain. A **crossing chain** is a chain whose 2 endpoints lie in different regions. Such a chain crosses through a wall from one region to another. The remaining chains (J, L, M, and N) are all crossing chains. Chains J and L cross between regions A and B. Chains M and N cross between regions B and C.

We make two observations about solvable link patterns:

1. **Every region must have an even number of crossing chains**.

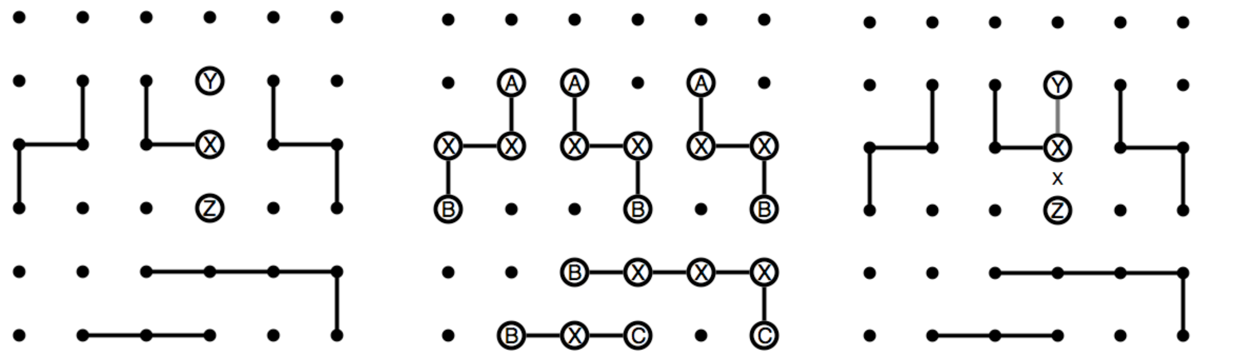
2. **Every region must contain an even number of endpoints**

These statements are actually equivalent, but let's see why they are true. The first must be true because the solution path has to cross into and out of the region an equal number of times, for a total number of crossings that is even. A region having an odd number of crossing paths is therefore another type of **forbidden pattern** and this can be used to rule out some links and force others as we will soon see.

The second formulation, in terms of endpoints, can be seen by observing that the **parity** (i.e, the even or oddness) of the number of endpoints is never changed as new links are added to the region. A link must either connect 2 open dots, 2 endpoints (of different chains), or an open dot and an endpoint. Connecting 2 open dots creates 2 new endpoints so the count goes up by 2, but the parity is the same. Connecting 2 endpoints fills each endpoint, reducing the count by 2, again preserving the parity. Finally, connecting an endpoint and an open dot will fill the endpoint but create a new endpoint out of the open dot, so the count is unchanged. Why must the count be even? If it were odd, then eventually, as the dots of the region get filled, the count would be reduced to 1, which means there is an isolated endpoint and that is forbidden. This is why a region with an odd number of endpoints is a forbidden pattern.

The equivalence of the two formulations can be seen by observing that every crossing chain belonging to a given region will contribute exactly 1 endpoint to that region. All other endpoints in the region must belong to internal chains, so they occur in pairs. Thus the parity of the number of crossing chains belonging to a region is the same as the parity of the total number of endpoints in the region (since they differ by an even number, either both counts are even, or both are odd).

Figure Applied Crossing Count illustrates how these concepts can be applied to rule out links and force others:



(a) Decision XY or XZ? **(b) Choice XY chains** **(c) Forbidden & forced links**

Figure: Applied Crossing Count

In part (a), we face a choice of whether to connect dot X to Y or Z. In part (b), we examine the consequences of connecting X to Z. This link fills dot X, and completes the separation of regions A and B. Now there are 3 crossing chains belonging to A, and it is forbidden to have an odd number of crossing chains. We could also count the endpoints. There are 3 endpoints in A and 5 endpoints in B, which is forbidden. Our conclusion must be that link X-Z is a forbidden link, since it leads to a forbidden pattern. In part (c) of the figure, we label this link with an X and observe that link X-Y is now forced.

Can a region have 0 crossing chains? The answer is no, as long as the region is not the only region. If there are 2 or more regions, they all must have crossing chains connecting them, otherwise it would be impossible to ever connect dots of one region with any dots of another (as is required in a legal solution).

Can a region have zero endpoints? Once again, the answer is no, as long there are 2 or more regions. If there are no endpoints, then there can be no crossing chains, since a crossing chain contributes one endpoint to each of the two regions it connects. Since zero crossing chains is forbidden (as we have just seen) it must be that a region (that is not the only region) with zero endpoints is also forbidden.

Summary of forbidden patterns involving regions

The following are all forbidden:

1. **A region with an odd number of crossing chains**

2. **A region with an odd number of endpoints**

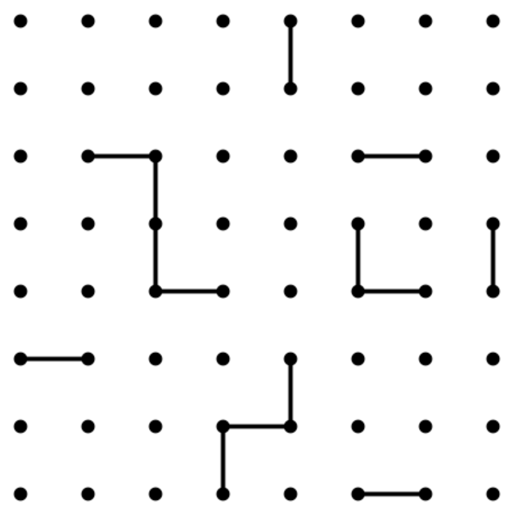
3. **A region with zero crossing chains (unless it is the only region)**

4. **A region with zero endpoints (unless it is the only region)**

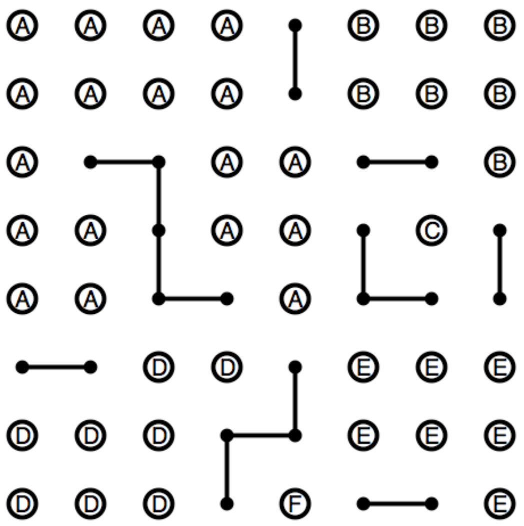
Remember that any link which would create or lead to a forbidden pattern is thus a forbidden link.

Dividing Regions into Sub-Regions

If we consider endpoints as well as filled dots, we can further divide regions into **sub-regions**, consisting of contiguous open dots. Two open dots are in the same sub-region if you can get from one dot to the other stepping neighbor to neighbor using only open dots. Two open dots are in different subregions if every path from one to the other (stepping from neighbor to neighbor) must necessarily go through an endpoint or a filled dot. Figure Sub-Regions illustrates this in a sample 8x8 puzzle.



Square 8x8 puzzle



Sub-Regions of square 8x8 puzzle

Figure Sub-Regions

A sample square 8x8 puzzle is shown on the left of the figure. On the right all of the open dots are labelled to indicate the subregions. There are six sub-regions in all. Regions C and F have only a single dot each.

Sub-regions lead to new forbidden patterns which can be useful for identifying forbidden and forced links. The next section describes some of these.

Forbidden patterns involving sub-regions

1. A sub-region with only 1 border endpoint is forbidden.

We say a dot **borders** a sub-region (or region) if it is not in the sub-region (or region) and it is a neighbor of some dot in the sub-region (or region). In the case of a sub-region with exactly 1 bordering endpoint, that endpoint needs an additional link, which when added will make it a filled dot. Then the sub-region becomes a region, since all its other border dots are already filled. There are two cases to consider, but both lead to forbidden patterns. If the lone endpoint links to an open dot internal to the sub-region, this creates a region with exactly one endpoint in it (the open dot linked to becomes this endpoint). There can't be any other endpoints since the sub-region had only 1 border endpoint to start with. But a region with an an odd number of endpoints is a forbidden pattern. In the other case, the lone border endpoint connects to a dot outside the subregion. This would turn the sub-region into a region of open dots having no endpoints. This is also a forbidden pattern. So no matter how we add a link to the endpoint, it leads to a forbidden pattern. Thus the original pattern with exactly 1 border endpoint is itself forbidden.

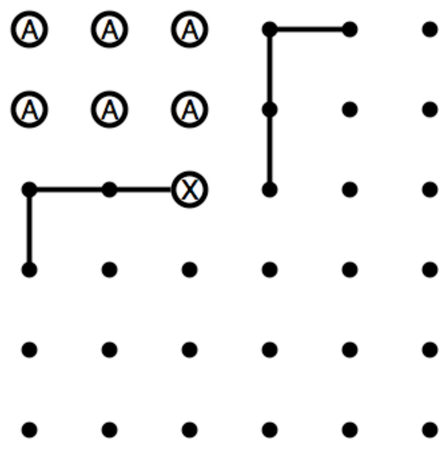


Figure: Sub-Region (A) with only 1 border endpoint (X)

Figure Sub-Region-1-border-endpoint illustrates this forbidden pattern. Sub-region A has exactly 1 bordering endpoint (X). Any link connecting to dot X will turn A into a region, and create a forbidden region pattern.

2. A chain that caps a sub-region is forbidden

We want to look closely at the endpoints that border a given sub-region. Each endpoint has a **matching** endpoint at the other end of the chain it belongs to. We define an endpoint to be **internal** (to the sub-region) if both it and its matching endpoint border the given sub-region and no other sub-region. Any other border endpoints of the sub-region are called **external**. There are 2 ways a bordering endpoint can be external: either by itself bordering on an additional sub-region, or by having its matching endpoint border a different sub-region.

We define a **capping chain of a sub-region** to be a chain whose 2 endpoints are the **only external endpoints** of the sub-region. This means that at least one of the endpoints of the capping chain must border on an additional sub-region (otherwise they would be internal endpoints). We say that such a chain **caps the sub-region** (it isolates it from all other sub-regions).

Why is a capping chain forbidden? Because no matter how we fill the 2 endpoints of the capping chain, we create a forbidden pattern. Observe that filling both endpoints of the capping chain will complete a wall defining a new region including all of the original sub-region plus its internal endpoints. The endpoints of the capping chain could each be filled by linking either into or away from the original subregion. If both endpoints connect the same way this does not create a crossing chain for the new region, and if they link in opposite ways (one into an the other away from the sub-region) then it becomes a single crossing chain contributing one endpoint to the new region. The key point is that there can be no other crossing chains for the new region, otherwise there would have been additional external endpoints of the original sub-region. But this leaves a region with either 0 or 1 crossing chains, both of which are forbidden. [Technically, to rule out the case of 0 crossing chains we must make sure that the new region is not the only region, but we know there were at least 2 sub-regions to start with because of the external endpoints of the capping chain. These 2 sub-regions cannot both be part of the same new region, so there must be at least two regions formed.]

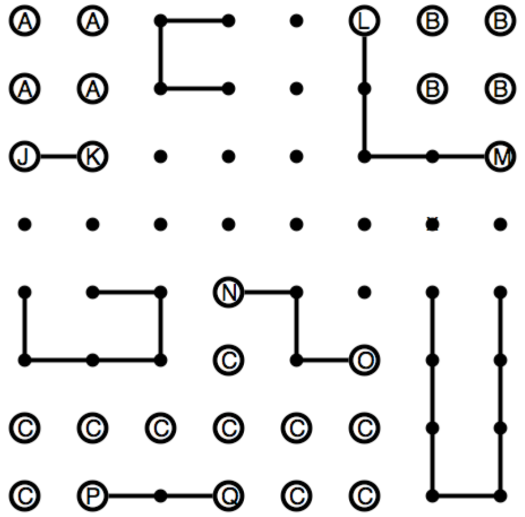


Figure Capping Chains

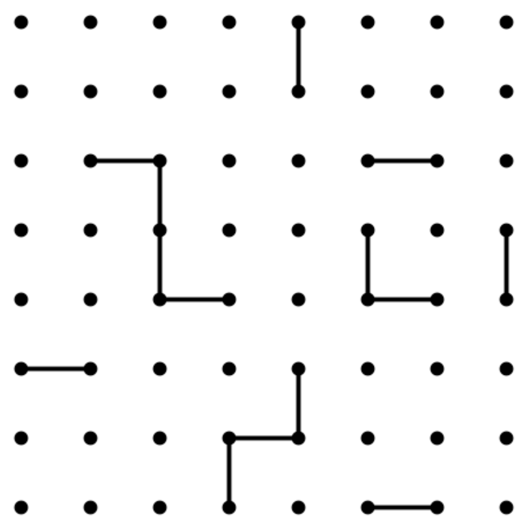
Figure Capping-Chains illustrates the forbidden patterns involving capping chains. Sub-region A is capped by the single-link chain J-K. Sub-region B is capped by the 4-link chain L-M. Region C is capped by the 3-link chain N-O. Note that the endpoints P-and Q are internal endpoints bordering sub-region C. In all cases, adding links to the capping chains creates forbidden region patterns, so these are all forbidden patterns themselves.

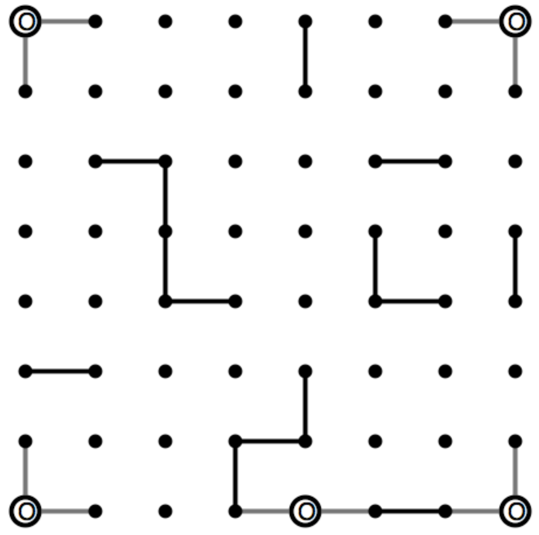
As in previous sections, these forbidden patterns can be used to identify forbidden links, which often leads to additional forced links. Any link that would create one of these (or any other) forbidden patterns, cannot be placed. That link is thus forbidden and can be ruled out.

SOLVING A COMPLETE PUZZLE USING THESE STRATEGIES

In this section we will walk through the complete solution of a square 8x8 puzzle. This will illustrate many of the concepts and strategies discussed above.

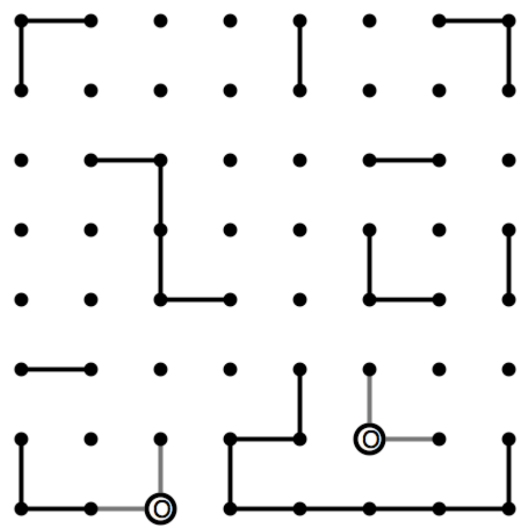
Figure **Start** presents the puzzle we will solve:


Start

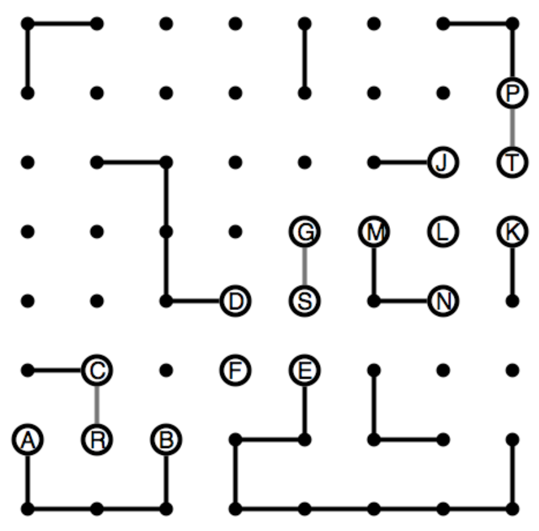


Step 1

In **Step 1** we look for simple forced links. There are 5 open dots (labelled O) which have only 2 available neighbors, so have 2 forced links at each of these dots. These forced links are shown in gray.

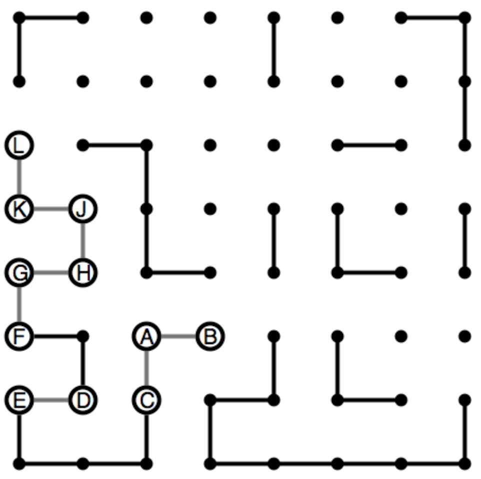


Step 2

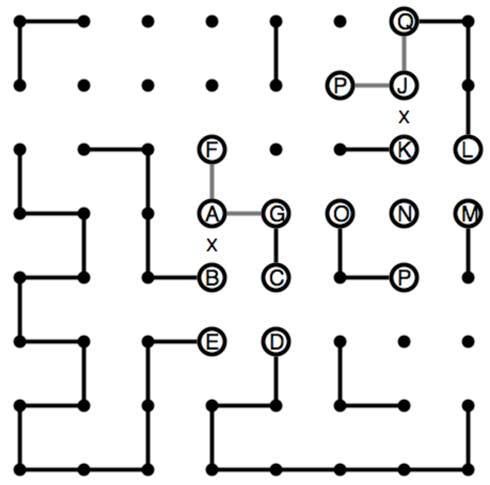


Step 3

In **Step 2** we now find two new open dots (labeled O) with just 2 available neighbors, so each of these has 2 forced links (shown in gray). In **Step 3** we employ the 3-way strategy at each of the dots R, S, and T. At dot R, the pair of links to A and B would create a forbidden short loop, so the third link R-C is forced. At dot S, the pair of links to D and E would isolate dot F. Since that is forbidden, the third link S-G is forced. At dot T, we employ 3-way with lookahead. The pair of links from T to J and K would force dot L to connect to both M and N creating a short loop, which is forbidden. We conclude that the T-J and T-K are a forbidden pair, so the third link T-P is forced. The forced links are shown in gray.



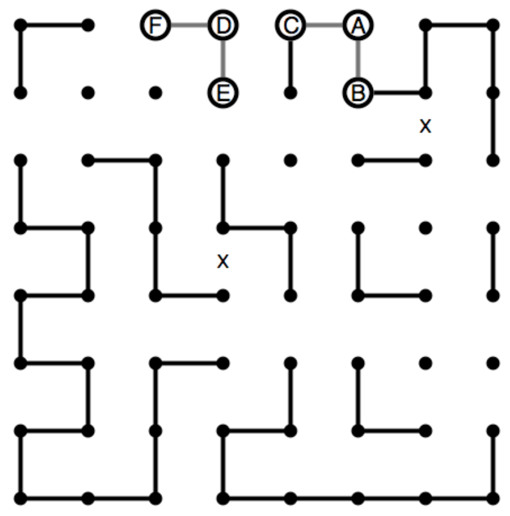
Step 4



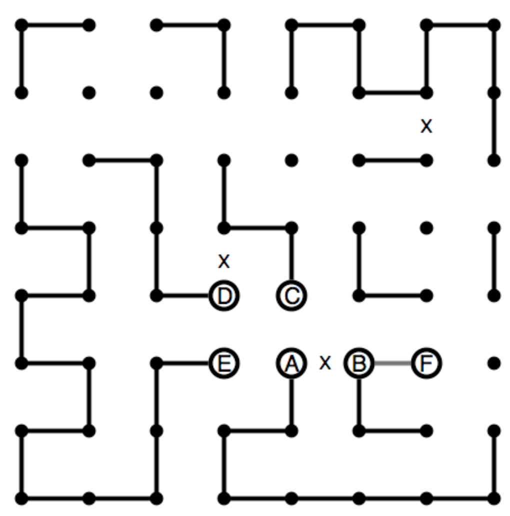
Step 5

In **step 4** we play out a sequence of simple forced links. We start at dot A, which now has only 2 available neighbors B and C, so links A-B and A-C are forced. Now dot C is an endpoint with just 1 available neighbor E, so this forces link D-E. Now that dot E is filled, endpoint can only be extended to G. Dot H must connect to both G and J, and finally, dot K is forced to link to J and L. All of these forced links are shown in gray.

In **step 5** we use lookahead to see that hypothetical links A-B and J-K each lead separately to forbidden patterns and can therefore be ruled out. Link A-B would fill dot B, forcing endpoint C to extend to D, filling D and thus isolating dot E (a forbidden pattern). So link A-B is forbidden and we mark it with an X. Now endpoint A is forced to extend to F. In the other case, link J-K would fill dot K and force endpoint L to connect to M. But now that K and M are filled, open dot N would have to connect to both O and P, which creates a forbidden short loop. We conclude that link J-K is forbidden, and mark it with an X. Now dot J is forced to connect to both Q and P.



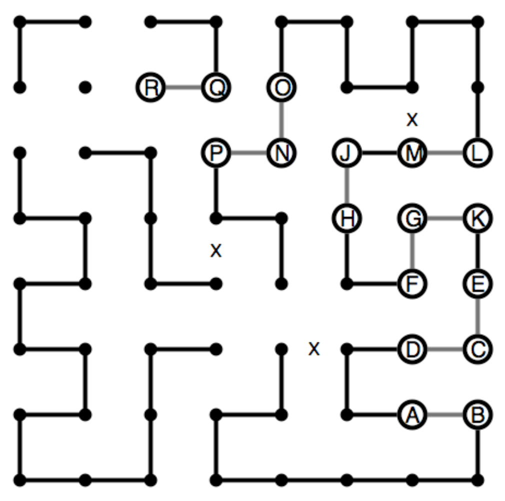
Step 6



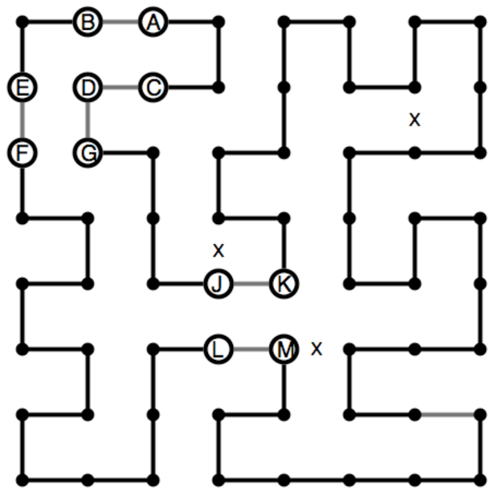
Step 7

In **Step 6** we add four more forced links. Dot A must connect to its only two available neighbors B and C. Then C is filled, so dot D must connect to its only 2 remaining available neighbors E and F.

In Step 7 it is hard to make progress without using Region Analysis. We consider what the possibility of link A-B, and observe that this link would create a region (dots C, D, and E) with 3 endpoints. Since an odd number of endpoints in a region is forbidden, we conclude that link A-B is a forbidden link, and mark it with X. Now endpoint B can only extend to F, so link B-F is forced.



Step 8

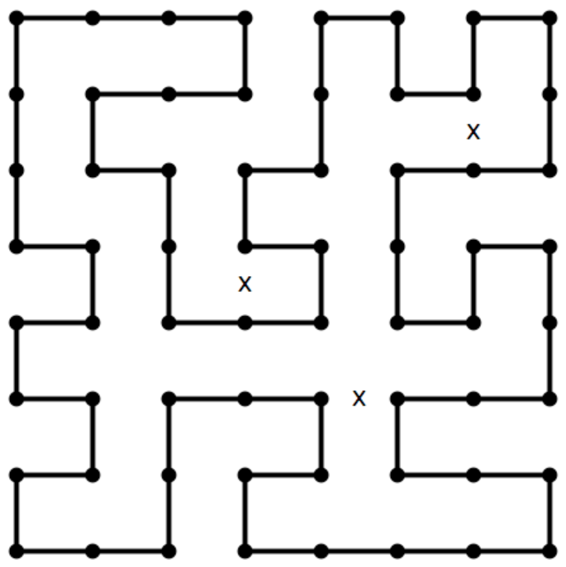


Step 9

In **Step 8** we follow out a sequence of forced links which nearly complete the solution. Endpoint A cannot connect to D (it would create a short loop), so link A-B is forced. Filling B forces C to connect to D and E. E is now filled so endpoint F must extend to its only available neighbor G. Endpoint H cannot connect to G (that would create a short loop), so it is forced to extend to J. At this point link G-M becomes forbidden since it would create a short loop, so G must extend to K and M must extend to L. With J filled, N is forced to connect to both O and P. Finally, since O and P are now filled, endpoint Q must extend to R. We are almost done!

In Step 9 we complete the puzzle with several more forced links. Link A-C is forbidden since it would create a short loop, so A must extend to B and C must extend to D. Now link E-D would close a short loop, so E must extend to F and D must extend to G. As our last step, we rule out link J-L (since it would create a short loop), thus forcing links J-K and L-M. These links close a loop, but it is not a short loop since it contains all the dots in the grid. It is a solution loop!

Here is the solution (with labels removed and all links drawn in black):



Because we only added links that were forced, we can be confident that this is in fact the only solution. The remainder of this book provides many puzzles to give you ample practice in applying the various reasoning strategies. Perhaps you will discover new strategies of your own. Enjoy!