Paper 164-2012

Formation of Optimum Accountable Care Organizations (ACOs) Leveraging SAS®/OR

Khasha Dehnad, AIMS Consulting, Holmdel, New Jersey, USA

ABSTRACT

Under the new health care reform law a network of doctors and hospitals, can form an “Accountable Care Organization (ACO)” and be awarded the responsibilities of managing and delivering all of health care needs of Medicare beneficiaries.

In this paper, we show how the rich language of Proc Optmodel and Patient-Level data can be used to facilitate the formation of optimum ACOs. Our methodology relies on identifying the optimum combination of social/referral networks of general practitioners, specialists and hospitals while meeting Medicare contractual requirements.

Representing practitioners as nodes and “common/referred patients between practitioners” as links, “proc optmodel” is utilized to calculate the shortest path between all nodes. Then links with the highest number of traversed shortest paths (betweeness) are iteratively removed in order to identify distinct sub communities of practitioners. Since different ordering of link removal results in different sub-networks, the process is repeated until the optimal network is discovered.

This paper describes the concept using a handful of nodes and links, then proceeds to show the application of the methodology to a large scale “real life” optimization problem.

INTRODUCTION

For many years, the growth in government and private Health Care spending has outpaced GDP growth. According to the 2005 congressional budget office, without a slowdown in Medicare and Medicaid expenditures, the net federal spending on the two programs would grow to 13 percent of GDP in 2040 and 38 percent of GDP by 2082

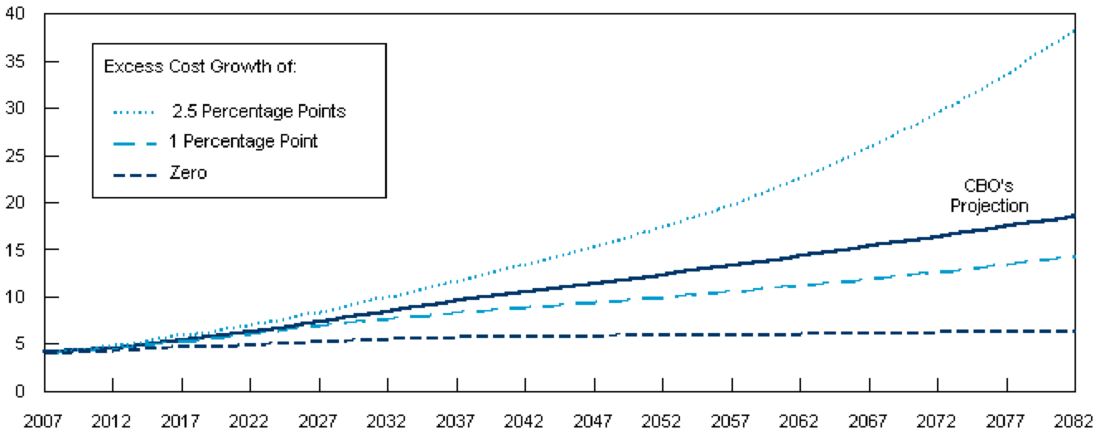


Figure 1. Source: Congressional Budget Office.

To curtail the rising cost of Medicare, under the a law, hospitals, doctors and other healthcare providers can form an Accountable Care Organization. Such an ACO contracts with the government, agrees to be for responsible for managing all of the health care needs of at least 5,000 Medicare beneficiaries for a minimum of three years.

Under the ACO model, healthcare providers receive bonuses based on their ability to meet healthcare standards, while reducing the government's per patient costs. Unlike Medicare's traditional fee-for-service, the profit incentives come from keeping patients healthy, and out of hospitals, as opposed to performing more tests and procedures.

While patients are assigned to ACOs, they are free to see doctors of their choice, in or out of the ACO network, without incurring additional fees. However it is in the best interest of the ACO to keep the patient within their own network, in order to best manage costs and provide quality healthcare.

As a result, it is optimal to form an ACO around existing "Networks of Care" for Medicare patients, rather than redirecting patients. This presents a challenge, and necessitates the identification of sub-networks. The following will show how to discover the network of care of Medicare patients, especially those with higher medical needs.

PATIENT DATA AND NETWORK OF CARE

Patient data plays a major role in the identification of Network of Care. Patient data is a collection of information obtained from patient encounters with health care providers. One of the sources of patient data is reimbursement data. Commercially available databases, capture, organize and store reimbursement claim data. The central feature of these databases is the anonymous patient, who is identified by a consistent (but otherwise meaningless) patient-ID.

Patient ID	Activity Date	Practitioner ID	Specialty Code	Specialty Description	Activity Description	Activity Code	Activity Type
12345	6-May-09	4351634	FM	FAMILY MEDICINE	INFECTIOUS AGENT, IMMUNOASSAY, DIRECT OB	87880	Prc
12345	2-Jun-09	4110365	PTH	PATHOLOGY, ANATOMIC/CLINICAL	MALIGNANT NEOPLASM OF THYROID GLAND	193	Diag
12345	3-Jun-09	1647226	GS	GENERAL SURGERY	MALIGNANT NEOPLASM OF THYROID GLAND	193	Diag
12345	6-Jun-09	4351634	FM	FAMILY MEDICINE	INFECTIOUS AGENT, IMMUNOASSAY, DIRECT OB	87880	Prc

Figure 2. Example: Anonymous Patient Level Data.

By establishing some simple business rules, one can very quickly extract a patient's sequence of care. For example, it appears that patient 12345 in the above chart has visited General Practitioner 4351634, Pathologist 4110365, Surgeon 1647226, and again General Practitioner 4351634. Representing each Practitioner as a node and each patient visit as an arc, the sequence of visits charted above can be represented graphically.

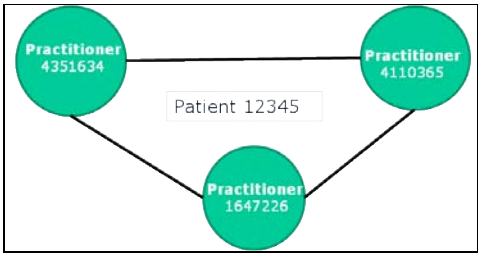


Figure 3. Example: Sequence of Patient 12345 Visits.

EXTRACTING SUB-NETWORKS AND IMPORTANT ARCS

At first, the graphical representation of the Network of Care for all patients can appear quite overwhelming. But in reality, the network consists of distinct and 'loosely' connected sub-networks. However, breaking the "loosely" connected sub-networks into distinct networks presents its own challenges.

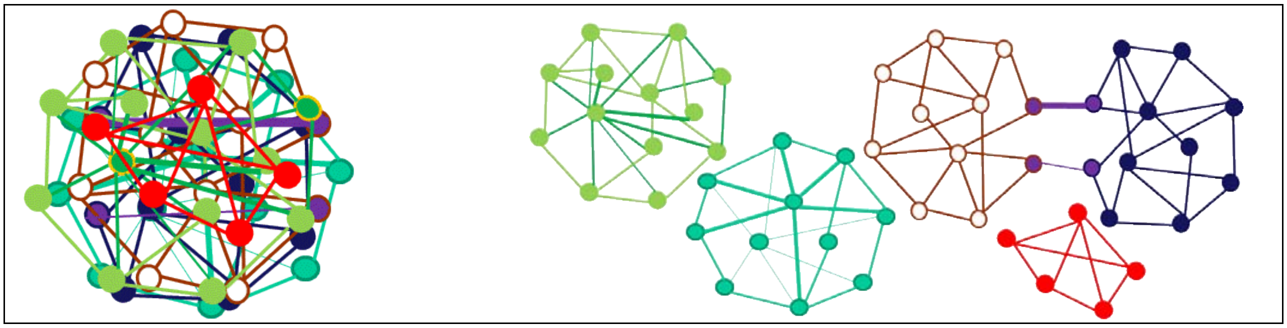


Figure 4. Example: Decomposition of the Overall Network.

In short, the idea is to identify the sub-networks and include, or exclude, additional nodes/providers so the contractual obligations are met and the potential for cost savings are maximized. For example, two distinct sub-networks can be combined to make sure that the ACO's minimum requirement of 5,000 patients is met. Similarly, a treatment center (one of the nodes) with fewer expected visits from the targeted population can be dropped, as long as the other facilities in the sub-network can meet government requirements (e.g. number of oncology treatment center per capita).

In essence, the problem comes down to identifying sub-networks and discovering the important arcs for inclusion or exclusion. Some of the most important arcs in the analysis are those that 'loosely' connect sub-networks, that is, those arcs, which when broken, split a sub-network into two or more distinct networks (arcs with the highest degree of 'betweeness'). The following, outlines the Optmodel programs used to implement these concepts (see Appendix for more details).

IDENTIFICATION OF SUB-NETWORKS

1. Choose a random node and add it to an empty set
2. Identify all the nodes that are reachable from the above set using the arcs.
3. Add the reachable nodes to the set and continue the process until no additional node is reachable. All the nodes in one Sub-Network have been identified.
4. Store the Sub-Network, empty the set, and continue the process until all the Sub-networks are identified.

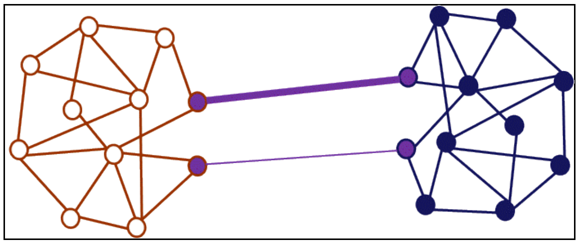


Figure 5: Example: Loosely connected sub-network

IDENTIFICATION OF ARCS CONNECTING TWO 'LOOSELY' CONNECTED SUB-NETWORKS

1. Calculate the shortest distance between every pair of nodes.
2. Identify the arcs that appear most often in all of the shortest paths.
3. Break one of the arcs (in case of a tie) in step 2, and continue until one node is not reachable from another (shortest path=∞).

Please note that the calculation for shortest paths between every pair of nodes translates into the redefinition of matrix multiplication, where addition takes the place of multiplication, and minimization takes the place of addition.

The SAS code in the appendix uses the "set" and the "logical" operator of Proc Optmodel to implement the above concepts. To make the example more understandable, the program is broken down into multiple Proc Optmodel steps and the length of each arc is set to 1. As a result, the programs are not written in the most efficient way and at times additional elements, or steps, are introduced to make the programs more explicit.

However in a real life implementation, the Optmodel steps are combined, and the length of each arc is set to a normalized number of patient visits. Furthermore, since patients with multiple co-morbidity represent the majority of the Medicare expenditure, the actual implementation includes only severely ill patients.

REAL LIFE EXAMPLE

The concepts above have been successfully applied to Patient-Level data, in order to enable the identification of an optimal Sub-Network of ACO Providers. The following is a graphical representation of one of the extracted ACO networks.

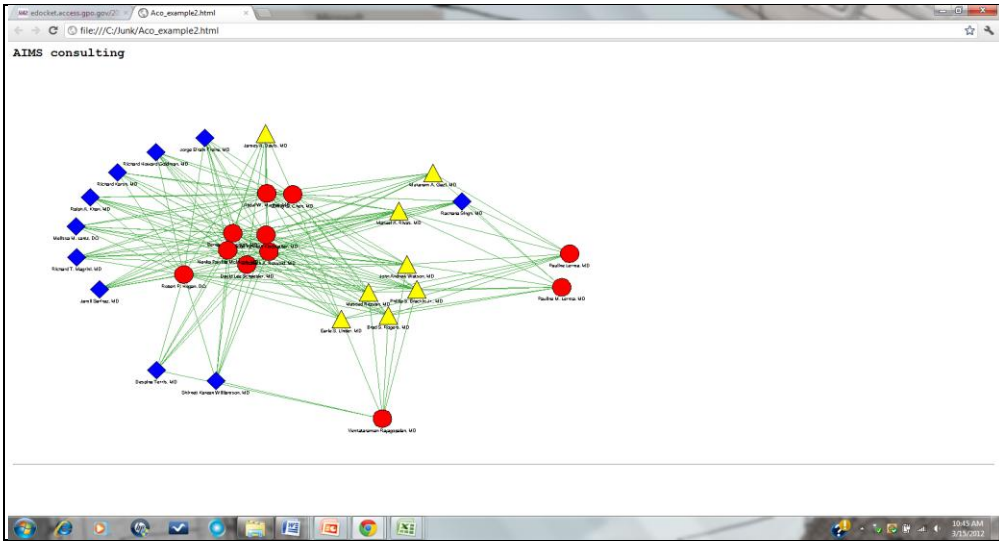


Figure 6: Example: Extracted ACO Networks

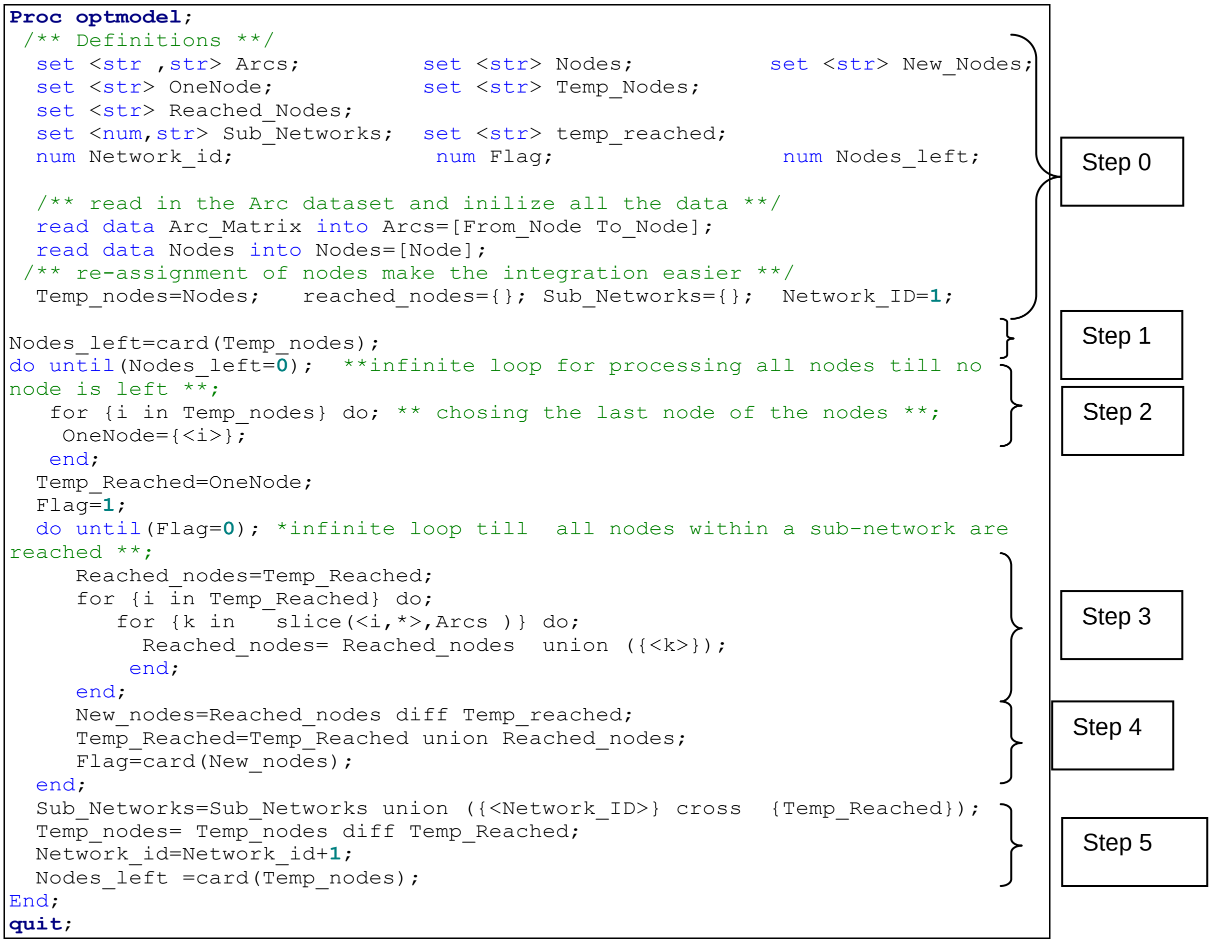
CONCLUSION

Proc Optmodel provides a rich 'set' language which opens the door for the easy implementation of sophisticated OR solutions. These solutions, such as the one in this paper, can be coded from scratch without substantial development expenses.

APPENDIX

IDENTIFICATION OF SUB-NETWORKS

- The following algorithm, implemented in Optmodel, extracts the sub-networks.
- Step 0: Define and initialize the sets and variables (e.g. set of 'Nodes', 'Arcs', 'Reached Nodes')
- Step 1: Check the cardinality of the 'Nodes' set. If empty, go to Step 6. Else go to step 2.
- Step 2: Select a random element of the 'Nodes' set (e.g. the last node) and add to the list of 'Reached Nodes'. Go to step 3.
- Step 3: Using the 'Arc' set, identify the 'New Nodes' that can be reached from the set of 'Reached Nodes'. If empty go to step 5 ; Else go to step 4.
- Step 4: Add the set of 'New Nodes' to the set of 'Reached Nodes'. Go to step 3.
- Step 5: Attach a Network ID to each element of the 'Reached Nodes' and store them in a separate set. Reduce the 'Nodes' set by the set of 'Reached Nodes'. Go to step 1
- Step 6: Stop. All nodes have been assigned to a network (implemented as a conditional loop).



```
Proc optmodel;
 /** Definitions **/
  set <str ,str> Arcs;          set <str> Nodes;          set <str> New_Nodes;
  set <str> OneNode;            set <str> Temp_Nodes;
  set <str> Reached_Nodes;
  set <num,str> Sub_Networks;   set <str> temp_reached;
  num Network_id;                num Flag;                  num Nodes_left;

  /** read in the Arc dataset and inilize all the data **/
  read data Arc_Matrix into Arcs=[From_Node To_Node];
  read data Nodes into Nodes=[Node];
 /** re-assignment of nodes make the integration easier **/
  Temp_nodes=Nodes;    reached_nodes={}; Sub_Networks={};  Network_ID=1;

Nodes_left=card(Temp_nodes);
do until(Nodes_left=0);   **infinite loop for processing all nodes till no
node is left **;
    for {i in Temp_nodes} do; ** chosing the last node of the nodes **;
     OneNode={<i>};
    end;
  Temp_Reached=OneNode;
  Flag=1;
  do until(Flag=0); *infinite loop till  all nodes within a sub-network are
reached **;
     Reached_nodes=Temp_Reached;
     for {i in Temp_Reached} do;
        for {k in   slice(<i,*>,Arcs )} do;
          Reached_nodes= Reached_nodes  union ({<k>});
         end;
     end;
     New_nodes=Reached_nodes diff Temp_reached;
     Temp_Reached=Temp_Reached union Reached_nodes;
     Flag=card(New_nodes);
  end;
  Sub_Networks=Sub_Networks union ({<Network_ID>} cross  {Temp_Reached});
  Temp_nodes= Temp_nodes diff Temp_Reached;
  Network_id=Network_id+1;
  Nodes_left =card(Temp_nodes);
End;
quit;
```

CALCULATION OF THE SHORTEST PATH BETWEEN ALL PAIRS OF NODES:

The challenge of breaking loosely linked networks into sub-networks is to identify the right link(s) to break. This can be accomplished by identifying the shortest path between all pairs of nodes. The arc that appears most often in the shortest paths (highest betweeness) acts as a bridge between two or more sub-networks. In the graph above labeled 'Example: Loosely Connected Sub-Network', the purple arcs have the highest betweeness, therefore they would be removed recursively until the two sub-networks are not connected.

Instead of computing the shortest paths from one pair of nodes at a time, we can use Proc Optmodel to develop a single integrated code that can calculate the shortest paths between all pairs of nodes in one step.

Let :

- Uij= the length of a shortest path from i to j
- Uij(m) the length off a shortest path from i to j, subject to the condition that the path contains no more than m arcs

If we consider the distance between each node and itself as zero (aij=0), then we have

- Uij(0) = 0 (i eq. j)
- Uij(0) = +∞ (i not eq. j)
- Uij(m=1)=mink{uik(m) + akj}

If we carry out the above computations, we obtain the shortest paths after n-1 (number of nodes -1) iterations.

The above computations can be represented as a new matrix manipulation, where addition takes the place of multiplication, and minimization takes the place of addition. In other words, the multiplication of matrix A and B would be defined as:

P={pij= mink{aik + bkj }} instead of P={pij=∑k aik bkj}

The following Proc Optmodel represents the computational details of the above calculation.

```
proc optmodel;
/** definitions **/
set <str , str > Arcs;   set <str> Nodes;
str PriorNode{Arcs};** last node in the shortest distance **;
num ArcLen{Arcs};        num Distance{Nodes,Nodes};
num Temp{Nodes,Nodes};   num row{Nodes};
num MinRow;              num CalcDist;
 /* initialization */
read data Arcs into Arcs=[From To] ArcLen  ;
Nodes=union{<i,j> in Arcs} {i} ;
for {i in Nodes, j in Nodes} do;
      PriorNode[i,j]=i;
      Distance[i,j]=ArcLen[i,j];
      Temp[i,j]= Distance[i,j] ;
end;

for {m in Nodes} do;
for {i in Nodes} do;
  for { j in Nodes} do;
  minrow=distance[i,j];
   for {k in Nodes} do;
     CalcDist=(Distance[i,k] + ArcLen[k,j]);
         if minrow<= CalcDist then do;  ** nothing **;end;
      else do;
         minrow=CalcDist;
        if (i ne k) and (j ne k) then do;
           PriorNode[i,j]=k;
         end;
      end;
   end;
          Temp[i,j]=minrow;

    end;
 end;
 ** reset the distance matrix **;
   for {i in Nodes, j in Nodes} do;
      Distance[i,j]=Temp[i,j];
   end;
end; ** end of calculations for shortest path **;
quit;
```

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name: Khasha Dehnad
Enterprise: AIMS Consulting
Address: 4 Taylor Run
City, State ZIP: Holmdel, New Jersey 07733
Work Phone: 732-357-6600
Fax: 732-957-0040
E-mail: kdehnad.aims@gmail.com
Web: www.aimsinfo.com