Global Inference Using Integer Linear Programming

Wen-tau Yih

August 15, 2004

1 Introduction

This report is a supplemental document of some of our papers [5, 3, 4]. It gives a simple but complete step-by-step case study, which demonstrates how we apply integer linear programming to solve a global inference problem in natural language processing. This framework first transforms an optimization problem into an integer linear program. The program can then be solved using existing numerical packages.

The goal here is to provide readers an easy-to-follow example to model their own problems in this framework. There are two main parts in this report. Sec. 2 describe a problem of labeling entities and relations simultaneously as our inference task. It then discusses the constraints among the labels and shows how the objective function and constraints are transformed to an integer linear program. Although transforming the constraints to their linear forms is not difficult in this entity and relation example, sometimes it can be tricky, especially when more variables are involved. Therefore, we discuss how to handle different types of constraints in Sec. 3.

2 Labeling Entities & Relations

Given a sentence, the task is to assign labels to the entities in this sentence, and identify the relation of each pair of these entities. Each entity is a phrase and we assume the boundaries of these entities are given.

Figure 1 gives an example of the sentence "Dole's wife, Elizabeth, is a native of N.C." In this sentence, there are three entities, *Dole*, *Elizabeth*, and *N.C.* We use E_1, E_2, and E_3 to represent their entity labels. In this example, possible entity labels include ***other***, ***person***, and ***location***. In addition, we would like to know the relation between each pair of the entities. For a pair of two entities E_i and E_j, the relation is represented by R_{ij}. In this example, there will be 6 relation variables – $R_{12}, R_{21}, R_{13}, R_{31}, R_{23}, R_{32}$. Since most entities have no special relation, the value of most relation variables should be ***irrelevant***. Besides this special label, the relations of interest in this example are ***spouse_of*** and ***born_in***.

Assume that magically some local classifiers have already provided some *confidence* scores on possible labels, as shown in Table 1.

If we want to choose the labels that maximize the sum of those confidence scores, it's the same as choosing the label that has the highest score for each variable. The global labeling then becomes:

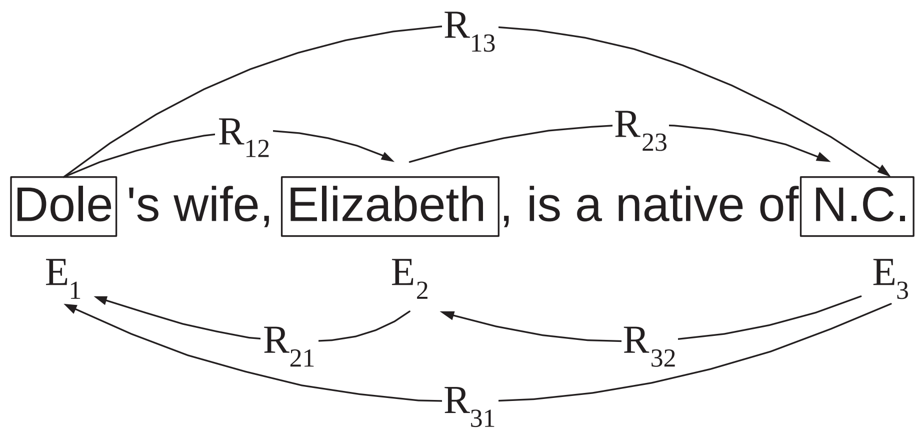

Figure 1: A sentence that has 3 entities

variable	other	person	location
E_1	0.05	0.85	0.10
E_2	0.10	0.60	0.30
E_3	0.05	0.50	0.45

variable	irrelevant	spouse_of	born_in
R_{12}	0.05	0.45	0.50
R_{21}	0.75	0.10	0.15
R_{13}	0.85	0.05	0.10
R_{31}	0.80	0.05	0.15
R_{23}	0.10	0.05	0.85
R_{32}	0.65	0.20	0.15

Table 1: The confidence scores on the labels of each variable.

variable	label	score
E_1	person	0.85
E_2	person	0.60
E_3	person	0.50
R_{12}	born_in	0.50
R_{21}	irrelevant	0.75
R_{13}	irrelevant	0.85
R_{31}	irrelevant	0.80
R_{23}	born_in	0.85
R_{32}	irrelevant	0.65
	sum	6.35

At this point, the problem seems to have been solved by the magic local classifiers. However, after a second look at this labeling, we can easily find the inconsistency between entity and relation labels. For example, R_{12} cannot be *born_in* if both entities E_1 and E_2 are *persons*. Indeed, there exists some natural constraints between the labeling of entity and relation variables that the local classifiers may not know or respect. In our example, we know the global labeling also stratifies the following two constraints.

- if R_{ij} = spouse_of, then E_i = person AND E_j = person
- if R_{ij} = born_in, then E_i = person AND E_j = location

In summary, the problem we want to solve here really is to find the best legitimate global labeling, which is subject to the constraints and maximizes the sum of the confidence scores.

Note that although exhaustive search seems plausible in this toy problem, it soon becomes intractable when the number of variables or the number of possible labels grows. In the rest of this section, we are going to show that how we transfer this problem to an integer linear program, and let the numerical packages help us to find the answer.

2.1 Indicator Variables

In order to apply (integer) linear programming, both the objective function and constraints have to be linear. Since the confidence score could be any real number, the original function is not linear. In addition, the logical constraints we have are not linear as well.

To overcome this difficulty, the first step of the transformation is to introduce several *indicator* (binary) variables, which represent the assignment of the original variables. For each entity or relation variable a and each legitimate label k, we introduce a binary variable $x_{a,k}$. When the original variable a is assigned label k, $x_{a,k}$ is set to 1. Otherwise, $x_{a,k}$ is 0. In our toy example, we then have 27 such indicator variables:

$$\begin{array}{ccc}
x_{E_1,\text{other}}, & x_{E_1,\text{person}} & x_{E_1,\text{location}}, \\
x_{E_2,\text{other}}, & x_{E_2,\text{person}}, & x_{E_2,\text{location}}, \\
x_{E_3,\text{other}}, & x_{E_3,\text{person}}, & x_{E_3,\text{location}}, \\
x_{R_{12},\text{irrelevant}}, & x_{R_{12},\text{spouse\_of}}, & x_{R_{12},\text{born\_in}}, \\
x_{R_{21},\text{irrelevant}}, & x_{R_{21},\text{spouse\_of}}, & x_{R_{21},\text{born\_in}}, \\
x_{R_{13},\text{irrelevant}}, & x_{R_{13},\text{spouse\_of}}, & x_{R_{13},\text{born\_in}}, \\
x_{R_{31},\text{irrelevant}}, & x_{R_{31},\text{spouse\_of}}, & x_{R_{31},\text{born\_in}}, \\
x_{R_{23},\text{irrelevant}}, & x_{R_{23},\text{spouse\_of}}, & x_{R_{23},\text{born\_in}}, \\
x_{R_{32},\text{irrelevant}}, & x_{R_{32},\text{spouse\_of}}, & x_{R_{32},\text{born\_in}}.
\end{array}$$

To simplify the notation, let $L_E = \{\text{other}, \text{person}, \text{location}\}$ and $L_R = \{\text{irrelevant}, \text{spouse\_of}, \text{born\_in}\}$ represent the sets of entity and relation labels, respectively. Assume $n = 3$ means the number of entities we have in the sentence. The indicator variables we introduce are:

$$\begin{array}{rl}
x_{E_i,l_e}, & \text{where } 1 \le i \le n \text{ and } l_e \in L_E \\
x_{R_{ij},l_r}, & \text{where } 1 \le i,j \le n, i \ne j, \text{ and } l_r \in L_R
\end{array}$$

2.2 Objective Function

Suppose c_{E_i,l_e} represents the confidence score of E_i being l_e, where $1 \le i \le n$ and $l_e \in L_E$, and c_{R_{ij},l_r} represents the confidence score of R_{ij} being l_r, where $1 \le i,j \le n, i \ne j$ and $l_r \in L_R$. The objective function f (i.e., the sum of confidence scores) can be represented by

$$f = \sum_{1 \le i \le n, l_e \in L_E} c_{E_i,l_e} x_{E_i,l_e} + \sum_{1 \le i,j \le n, i \ne j, l_r \in L_R} c_{R_{ij},l_r} x_{R_{ij},l_r}$$

If we plug in the numbers in Table 1, the function f is:

$$f = 0.05 \cdot x_{E_1,\text{other}} + 0.85 \cdot x_{E_1,\text{person}} + \cdots + 0.65 \cdot x_{R_{32},\text{irrelevant}} + 0.20 \cdot x_{R_{32},\text{spouse\_of}} + 0.15 \cdot x_{R_{32},\text{born\_in}}$$

Inevitably, this transformation also brings new constraints, which come from the fact that one entity/relation variable can only have one label, and must have one label. For example, only exact one of the labels *other*, *person*, *location* can be assigned to E_1. As a result, only one of the indicator variables $x_{E_1,other}, x_{E_1,person}, x_{E_1,location}$ can and must be 1. This restriction can be easily written as the following linear equations.

$$\begin{aligned} \textstyle\sum_{l_e \in L_E} x_{E_i,l_e} &= 1 \quad \forall 1 \le i \le n \\ \textstyle\sum_{l_r \in L_R} x_{R_{ij},l_r} &= 1 \quad \forall 1 \le i,j \le n, i \ne j \end{aligned}$$

2.3 Logical Constraints

The other reason of introducing indicator variables is to handle the real constraints we have – the logical constraints between entity and relation labels. Let me remind you what they are in our example:

- if R_{ij} = spouse_of, then E_i = person AND E_j = person, where $1 \le i,j \le n$ and $i \ne j$
- if R_{ij} = born_in, then E_i = person AND E_j = location, where $1 \le i,j \le n$ and $i \ne j$

If we treat the indicator variables as boolean variables, where 1 means *true* and 0 means f*alse*, the constraints can be rephrased as:

$$\begin{aligned} x_{R_{ij},\text{spouse\_of}} \rightarrow x_{E_i,person} \wedge x_{E_j,person} &\quad 1 \le i,j \le n, \text{ and } i \ne j \\ x_{R_{ij},born\_in} \rightarrow x_{E_i,person} \wedge x_{E_j,location} &\quad 1 \le i,j \le n, \text{ and } i \ne j \end{aligned}$$

In fact, these two boolean constraints can be modeled by the following two linear inequalities.

$$\begin{aligned} 2 \cdot x_{R_{ij},\text{spouse\_of}} \le x_{E_i,\text{person}} + x_{E_j,\text{person}} &\quad 1 \le i,j \le n, \text{ and } i \ne j \\ 2 \cdot x_{R_{ij},\text{born\_in}} \le x_{E_i,\text{person}} + x_{E_j,\text{location}} &\quad 1 \le i,j \le n, \text{ and } i \ne j \end{aligned}$$

Let's do a simple check to see if they are correct. When $x_{R_{ij},\text{spouse\_of}}$ is 0 (false), $x_{E_i,\text{person}}$ and $x_{E_j,\text{person}}$ can be either 0 or 1, and the inequality still holds. However, when $x_{R_{ij},\text{spouse\_of}}$ is 1 (true), both $x_{E_i,\text{person}}$ and $x_{E_j,\text{person}}$ have to be 1 (true).

Transforming the logical constraints into linear forms is the key of this framework. It is not hard, but may be tricky sometimes (which makes it an interesting brain exercise). We will talk more about transforming other types of logical constraints in Sec. 3 later.

2.4 Solving the Integer Linear Program Using Xpress-MP

Figure 2 shows the complete integer linear program. Now, all we need to do now is to apply some numeric packages, such as Xpress-MP [7], CPlex [1], or the LP solver in R [6], to solve it. Transferring the solution back to the global labeling we want is straightforward – just find those indicator variables that have the value 1. In this section, I will demonstrate how to apply Xpress-MP to do the job.

The syntax in Xpress-MP is fairly easy and straightforward. Here I simply list the source code with some comments, which are the lines beginning with the "!" symbol.

$$\max \quad \sum_{1\le i\le n, l_e\in L_E} c_{E_i,l_e} x_{E_i,l_e} + \sum_{1\le i,j\le n, i\ne j, l_r\in L_R} c_{R_{ij},l_r} x_{R_{ij},l_r}$$

subject to:

$$x_{E_i,l_e} \in \{0,1\} \qquad \forall 1 \le i \le n \qquad (1)$$

$$x_{R_{ij},l_r} \in \{0,1\} \qquad \forall 1 \le i,j \le n, i \ne j \qquad (2)$$

$$\sum_{l_e\in L_E} x_{E_i,l_e} = 1 \qquad \forall 1 \le i \le n \qquad (3)$$

$$\sum_{l_r\in L_R} x_{R_{ij},l_r} = 1 \qquad \forall 1 \le i,j \le n, i \ne j \qquad (4)$$

$$2 \cdot x_{R_{ij},\text{spouse\_of}} \le x_{E_i,\text{person}} + x_{E_j,\text{person}} \qquad 1 \le i,j \le n, \text{ and } i \ne j \qquad (5)$$

$$2 \cdot x_{R_{ij},\text{born\_in}} \le x_{E_i,\text{person}} + x_{E_j,\text{location}} \qquad 1 \le i,j \le n, \text{ and } i \ne j \qquad (6)$$

Figure 2: The complete integer linear program

```
model "Entity Relation Inference"
 uses "mmxprs"

parameters
  DATAFILE = "er.dat"
  Num_Entities = 3;
end-parameters

declarations
  ENTITIES = 1..Num_Entities
  ENT_CLASSES = {"Other", "Person", "Location"}
  REL_CLASSES = {"Irrelevant", "SpouseOf", "BornIn"}

  scoreEnt: array(ENTITIES, ENT_CLASSES) of real
  scoreRel: array(ENTITIES, ENTITIES, REL_CLASSES) of real
end-declarations

! DATAFILE stores the confidence scores from the local classifiers.
initializations from DATAFILE
  scoreEnt   scoreRel
end-initializations

! These are the indicator variables. declarations
  ent : array(ENTITIES, ENT_CLASSES) of mpvar
  rel : array(ENTITIES, ENTITIES, REL_CLASSES) of mpvar
end-declarations

! The objective function: sum of confidence scores
Obj := sum(u in ENTITIES, e in ENT_CLASSES) scoreEnt(u,e)*ent(u,e)
     + sum(u,v in ENTITIES, r in REL_CLASSES | u <> v) scoreRel(u,v,r)*rel(u,v,r)
```

```
! Constraints (1) and (2): the indicator variables take only binary values
forall(u in ENTITIES, e in ENT_CLASSES)
  ent(u,e) is_binary
forall(e1,e2 in ENTITIES, r in REL_CLASSES | e1 <> e2)
  rel(e1,e2,r) is_binary

! Constraints (3) and (4): sum = 1
forall(u in ENTITIES) sum(e in ENT_CLASSES)
  ent(u,e) = 1
forall(u,v in ENTITIES | u <> v) sum(r in REL_CLASSES)
  rel(u,v,r) = 1

! Constraints (5) and (6): logical constraints on entity and relation labels
forall(e1,e2 in ENTITIES | e1 <> e2)
  2*rel(e1,e2,"SpouseOf") <= ent(e1,"Person") + ent(e2,"Person")
forall(e1,e2 in ENTITIES | e1 <> e2)
  2*rel(e1,e2,"BornIn") <= ent(e1,"Person") + ent(e2,"Location")

! Solve the problem
maximize(Obj)

! Output the indicator variables that are 1
forall(u in ENTITIES, e in ENT_CLASSES | getsol(ent(u,e)) >= 1)
  writeln(u, " ", e)
forall(e1,e2 in ENTITIES, r in REL_CLASSES | e1 <> e2 and getsol(rel(e1,e2,r)) >= 1)
  writeln(e1, " ", e2, " ", r)

end-model
```

3 Transforming Logical Constraints into Linear Forms

This section summarizes and revises some rules of transforming logical constraints to linear (in)equalities described in [2]. To simplify the illustration, symbols a, b, c and $x_1, x_2, \cdots, x_n$ are used to represent indicator variables, which are treated as boolean variables and binary variables at the same. As usual, the values $0, 1$ represents the truth values *false* and *true*, respectively.

3.1 Choice Among Several Possibilities

In our entity and relation example, we have already processed the constraint "exactly k variables among $x_1, x_2, \cdots, x_n$ are true", where $k = 1$. The general form of this linear equation is:

$$x_1 + x_2 + \cdots + x_n = k$$

Another constraint, "at most k variables among $x_1, x_2, \cdots, x_n$ can be true", can be represented in a similar inequality.

$$x_1 + x_2 + \cdots + x_n \leq k$$

Uninterestingly, "k or more variables among $x_1, x_2, \cdots, x_n$ must be true" will be

$$x_1 + x_2 + \cdots + x_n \geq k$$

3.2 Implications

Implications are usually the logical constraints we encounter. While handling two or three variables may be trivial, extending it to more variables may be tricky. Here we illustrate how to develop the ideas from the simplest case to complicated constraints.

Two variables Suppose there are only two indicator variables a, b in the implication. The constraint, $a \rightarrow b$, can be represented as $a \leq b$. This can be easily verified by the following truth table.

$a \leq b$	$b = 0$	$b = 1$
$a = 0$	true	true
$a = 1$	false	true

What if we need to deal with something like $a \rightarrow \bar{b}$? The value of the compliment of b is exactly $1 - b$. Therefore, the corresponding linear constraint is $a \leq 1 - b$, or $a + b \leq 1$.

The relation "if and only if" is straightforward too. $a \leftrightarrow b$ is identical to $a \rightarrow b$ and $b \rightarrow a$. The corresponding linear constraints are $a \leq b$ and $b \leq a$, which is in fact $a = b$.

Three variables Now, let's try to generalize the implication a little bit to cover three variables. Since $a \rightarrow b \wedge c$ can be separated as $a \rightarrow b$ and $a \rightarrow c$, the straightforward transformation is to put two linear inequalities $a \leq b$ and $a \leq c$. Alternatively, the transformation in our entity and relation example "$2a \leq b + c$" also suffice, which is easy to check using a truth table.

Another implication, $a \rightarrow b \vee c$, can be modeled by $a \leq b + c$. This is because when $a = 1$, at least one of b and c has to be 1 to make the inequality correct.

What about the inverse of the above two implications? They can be derived using the compliment and DeMorgan's Theorem. $b \wedge c \rightarrow a$ is equivalent to $\bar{a} \rightarrow \overline{b \wedge c}$, which is $\bar{a} \rightarrow \bar{b} \vee \bar{c}$. Use the above rule and the the compliment, it can be modeled by $(1-a) \leq (1-b) + (1-c)$, or $a \geq b + c - 1$. $b \vee c \rightarrow a$ is equivalent to $b \rightarrow a$ and $c \rightarrow a$, so it can be modeled by two inequalities $b \leq a$ and $c \leq a$. Alternatively, this can be transformed to $\bar{a} \rightarrow \overline{b \vee c}$, which is $\bar{a} \rightarrow \bar{b} \wedge \bar{c}$. Therefore, it can be modeled by $2(1-a) \leq (1-b) + (1-c)$, or $\frac{b+c}{2} \leq a$.

More variables A logical constraint that has more variables can be complicated. Therefore, we only discuss some common cases here. Suppose we want to model "if a, then k or more variables among $x_1, x_2, \cdots, x_n$ are true." We can extend the transformation of $a \rightarrow b \vee c$, and use the following linear inequality.

$$a \leq \frac{x_1 + x_2 + \cdots + x_n}{k}$$

This transforation is certainly valid for $k = 1$. It is also easy to verify for other cases. If $a = 0$, then the right-hand-side is always larger or equal to 0, and the inequality is satisfied. However, when $a = 1$, it forces at least k x's are true, which is exactly what we want.

The next case we would like to try is the inverse, which is "if k or more variables among $x_1, x_2, \cdots, x_n$ are true, then a is true." This might be somewhat tricker than others. Our first guess might be:

$$(x_1 + x_2 + \cdots + x_n) - (k - 1) \leq a$$

Original constraint	Linear form
Exactly k of $x_1, x_2, \cdots, x_n$	$x_1 + x_2 + \cdots + x_n = k$
At most k of $x_1, x_2, \cdots, x_n$	$x_1 + x_2 + \cdots + x_n \leq k$
At least k of $x_1, x_2, \cdots, x_n$	$x_1 + x_2 + \cdots + x_n \geq k$
$a \rightarrow b$	$a \leq b$
$a = \bar{b}$	$a = 1 - b$
$a \rightarrow \bar{b}$	$a + b \leq 1$
$\bar{a} \rightarrow b$	$a + b \geq 1$
$a \leftrightarrow b$	$a = b$
$a \rightarrow b \wedge c$	$a \leq b$ and $a \leq c$
	or, $a \leq \frac{b+c}{2}$
$a \rightarrow b \vee c$	$a \leq b + c$
$b \wedge c \rightarrow a$	$a \geq b + c - 1$
$b \vee c \rightarrow a$	$a \geq \frac{b+c}{2}$
if a then at least k of $x_1, x_2, \cdots, x_n$	$a \leq \frac{x_1+x_2+\cdots+x_n}{k}$
if at least k of $x_1, x_2, \cdots, x_n$ then a	$a \geq \frac{x_1+x_2+\cdots+w_n-(k-1)}{n-(k-1)}$
$a = x_1 \cdot x_2 \cdots x_n$	$a \leq \frac{x_1+x_2+\cdots+x_n}{n}$ and $a \geq x_1 + x_2 + \cdots + x_n - (n-1)$

Table 2: Rules of mapping constraints to linear (in)equalities

Although this may seem correct at the first glance, we observe that the left-hand-side (LHS) will be larger than 1 when more than k of the x variables are 1. Because a can be either 0 or 1, this constraint will be infeasible. In fact, what we really need is to *squash* the LHS to less than 1. Currently, the largest possible value of the left-hand-side is $n - (k - 1)$. Therefore, dividing the LHS by $n - (k - 1)$ should suffice.

$$\frac{(x_1 + x_2 + \cdots + x_n) - (k-1)}{n - (k-1)} \leq a$$

Let's examine two special cases of this transformation to see if they are correct. Remember $b \vee c \rightarrow a$ is indeed one of these cases, given that $n = 2$ and $k = 1$. The linear inequality $\frac{b+c}{2} \leq a$ is exactly the same as what we derived previously. The other special case is "$x_1 \wedge x_2 \wedge \cdots \wedge x_n \rightarrow a$", which is equivalent to say $k = n$ here. Obviously, $a \geq x_1 + x_2 + \cdots + w_n - (n-1)$ is correct. One interesting observation is that the conjunction of a set of boolean variables is the same as the product of the corresponding binary variables. Therefore, the nonlinear constraint $a = x_1 \cdot x_2 \cdots x_n$ is the same as $a = x_1 \wedge x_2 \wedge \cdots \wedge x_n$. Its linear transformation is therefore $a \geq x_1 + x_2 + \cdots + x_n - (n-1)$ and $a \leq \frac{x_1+x_2+\cdots+x_n}{n}$.

Table 2 summarizes all the transformations we have discussed in this section.

4 Conclusions

Thanks to the theoretical developments of integer linear programming in the last two decades, and the tremendous improvement on hardware and software technology, numerical packages these days are able to solve many integer linear programming problems within very short time, even though ILP is in general NP-hard.

In this report, we have provided an entity and relation problem as example, and discussed several cases for transforming boolean constraints. We hope these illustrations are helpful to remodeling your inference problem, and allow you to take advantage of the numerical LP solvers as well.

References

[1] CPLEX. ILOG, Inc. CPLEX. http://www.ilog.com/products/cplex/, 2003.

[2] C. Guéret, C. Prins, and M. Sevaux. *Applications of optimization with Xpress-MP*. Dash Optimization, 2002. Translated and revised by Susanne Heipcke.

[3] V. Punyakanok, D. Roth, W. Yih, and D. Zimak. Semantic role labeling via integer linear programming inference. In *Proceedings of COLING 2004*, 2004.

[4] V. Punyakanok, D. Roth, W. Yih, D. Zimak, and Y. Tu. Semantic role labeling via generalized inference over classifiers. In *Proceedings of CoNLL 2004*, 2004.

[5] D. Roth and W. Yih. A linear programming formulation for global inference in natural language tasks. In *Proceedings of CoNLL-2004*, pages 1–8, 2004.

[6] The R Project for Statistical Computing. http://www.r-project.org/, 2004.

[7] Xpress-MP. Dash Optimization. Xpress-MP. http://www.dashoptimization.com/products.html, 2003.