Segmentation to more than 2 classes

- The flow-cut equivalence no longer works.
- Assume we have r classes.
  - encode class at each pixel with a one-hot vector

label at ij $\quad \underline{l}_{ij} = \underbrace{[0 \ \cdots \ 1 \ \cdots \ 0]}$

r d vector

1 element $= 1$ others 0

- Notice

$$\underline{l}_{ij}^T \, 1 = 1$$

$$\underline{l}_{ij} \in \{0,1\}^r$$

we have some form of prob model per class, and use the same kind of smoothing.

So:

$\underline{\min}$ linear term in $[\underline{l}_{ij}]$ + Quad term in $[\underline{l}_{ij}]$

$\underline{st}$ $\underline{l}_{ij}^T \mathbf{1} = 1$

$\underline{l}_{ij} \in \{0,1\}^r$

I: this is easily turned into a linear program

II: this linear program does not have tractable constraints
- we can't get a TUM problem out of it

III: solving problems of this kind is useful

I: turning into a linear program

(lose an index on labels)

$\underline{l}_a(i) = \{\, i\text{'th component of } \underline{l}_a \,\}$

$q_{ab}(i,j) = \{\, i,j\text{'th component of } \underline{q}_{ab} \,\}$

q_{ab} is an $r \times r$ table that represents the product $l_a \times l_b$

$$q_{ab}(i,j) = \begin{cases} 1 & \text{if } l_a(i) = 1 \text{ and } l_b(j) = 1 \\ 0 & \text{otherwise} \end{cases}$$

Note: $q_{ab}(i,j) \in \{0,1\}^{r \times r}$

$\underline{1}^T q_{ab} \underline{1} = 1$ (theres only one one in q_{ab})

Then

$$q_{ab}(i,\cdot) \leq l_a(i)$$

$$q_{ab}(\cdot,j) \leq l_b(j)$$

$$q_{ab}(i,j) \geq l_a(i) + l_b(j) - 1$$

- Straighten q_{ab} into a vector
  - stack these vectors into $\underline{q}$
  - stack l_a into a vector $\underline{l}$
- get:

$$\min \quad a^T \underline{l} + b^T \underline{q}$$

$$\text{st} \quad A\underline{l} + B\underline{q} = \underline{c} \qquad Ml + Nq \geq w$$

$$l \in \{0,1\}^{r \times N_p}$$

$$q \in \{0,1\}^{r \times r \times N_p} \longleftarrow \quad N_p = \#\text{ of pixels}, \quad r = \#\text{ of labels}$$

The linear constraints aren't TUM.

- recall $\ell_a^T 1 = 1$
- can't drop

$$q_{ab}(i,j) \geq \ell_a(i) + \ell_b(j) - 1$$

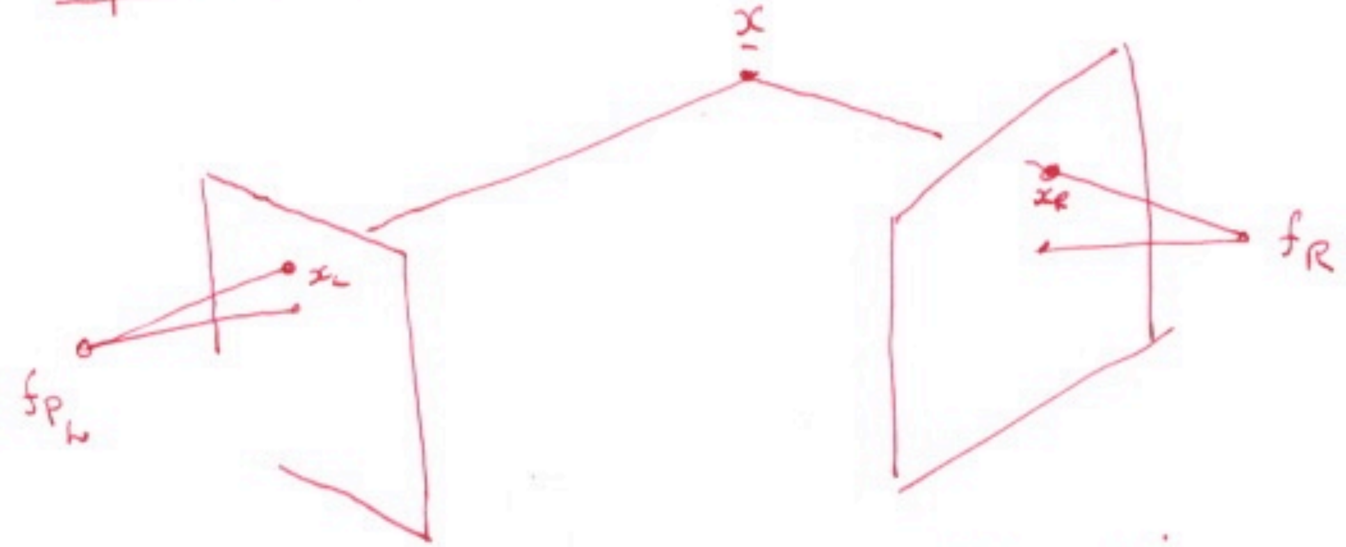
or

$$1^T q_{ab} 1 = 1$$

- Now what?

III : solving problems like this is useful.

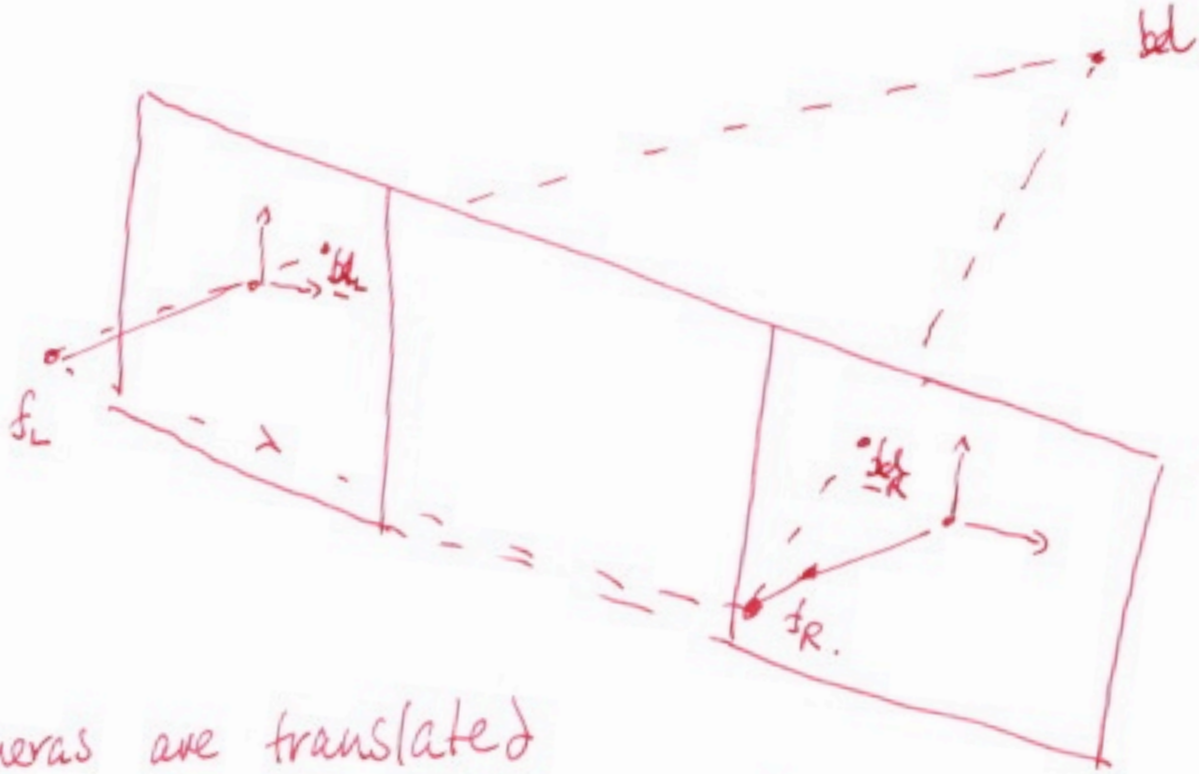
- depth estimation



if I know:
- camera configuration
- x_L and x_R

can recover $\underline{x}$
(and so distance to $\underline{x}$)

We work with a standardized geometry (it is known how to map to this from others)



· Cameras are <u>translated</u>

· parallel to image plane

· so image planes ~~have t~~ are the same plane.

· focal lengths are the same

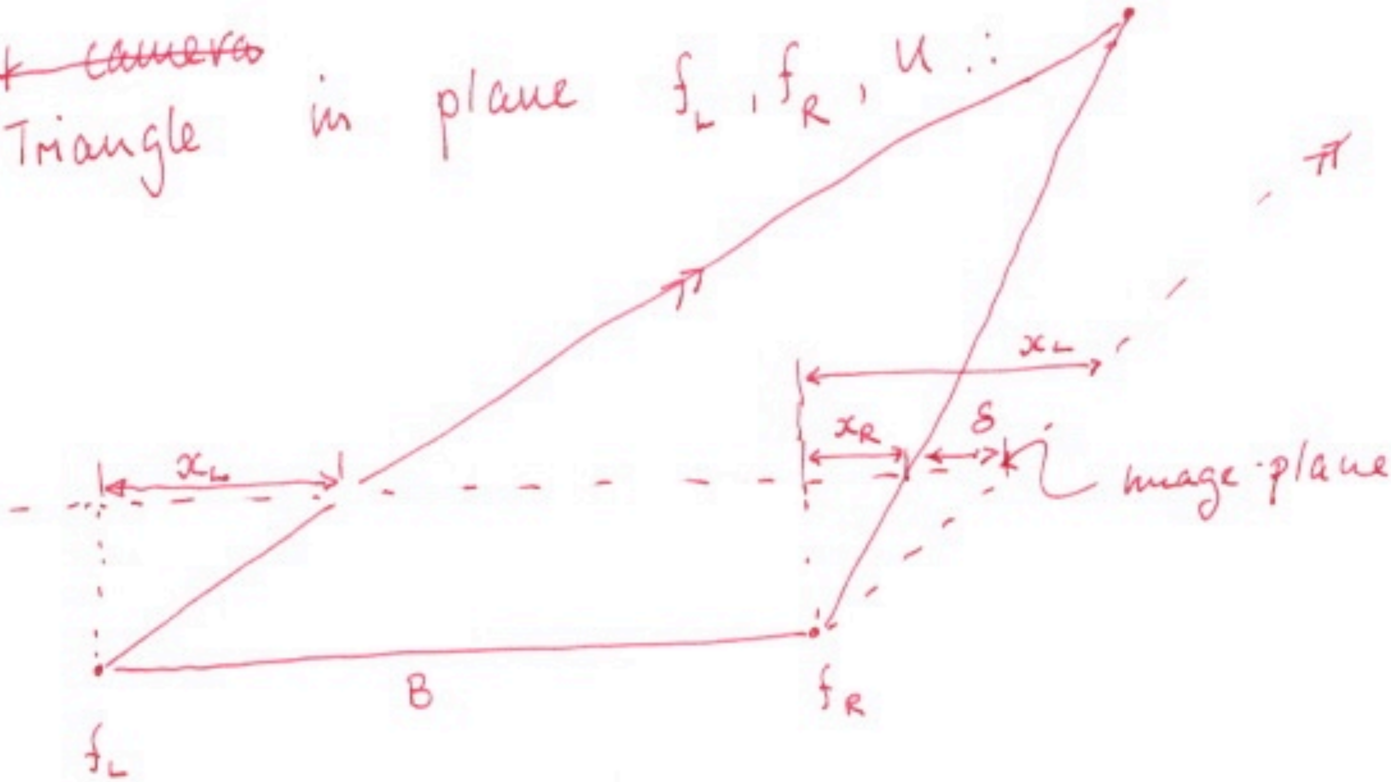
· coordinate system at camera center

$u_L = (x_L, y_L)$
$u_R = (x_R, y_R)$

It follows from drawing that: $y_L = y_R$.

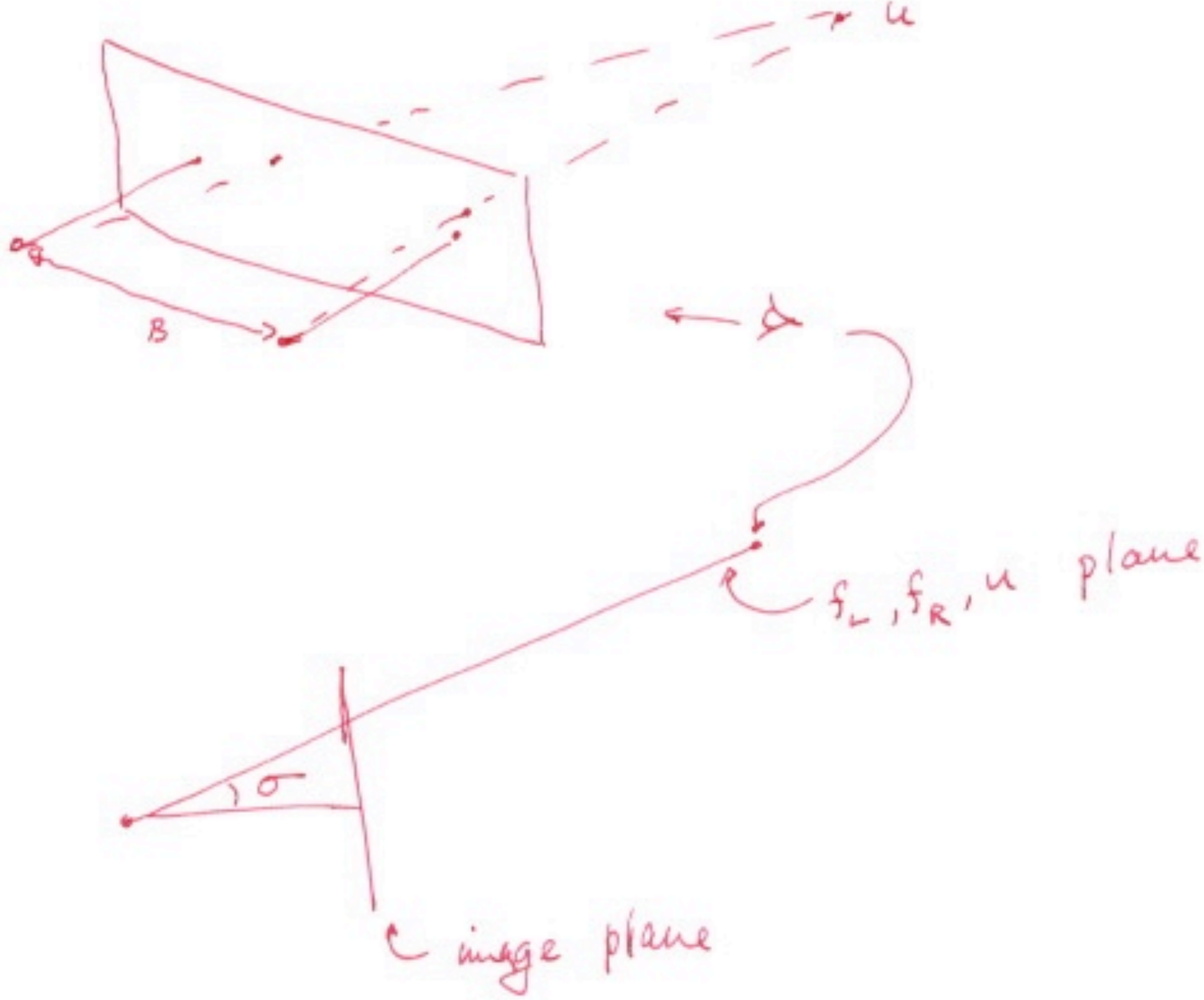
<u>But</u> x_L is <u>not</u> x_R

~~Left camera~~
Triangle in plane f_L, f_R, u:

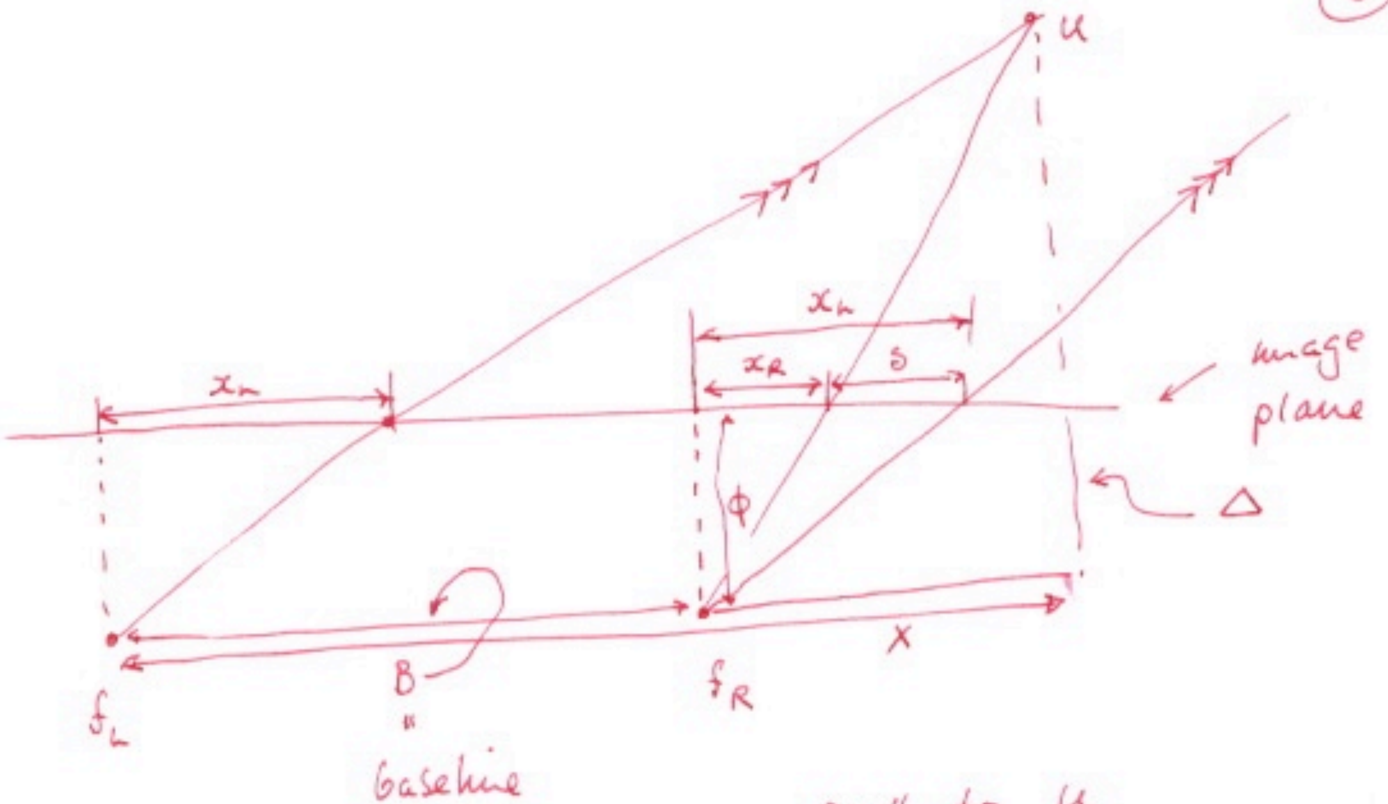


<u>NOTICE</u>: if I know x_L, y_L, δ
. I know u in 3D.

. in fact, $\delta \propto \frac{1}{Z_u}$ ← depth to u from camera plane.



- Place origin at f_L; image plane is $z = 1$; then picture on f_L, f_R, u plane is



depth to u

Now $$\Delta = \frac{Z}{\cos\sigma}$$

$$\phi = \frac{f}{\cos\sigma}$$

Similar triangles yield: $$x_L = \frac{\phi X}{\Delta} = \frac{fX}{Z}$$

$$x_R = \frac{\phi(X-B)}{\Delta} = \frac{f(X-B)}{Z}$$

So $$x_L - x_R = \frac{fB}{Z}$$

Now assume u has the same <u>color</u> in each camera. (good, unless u is specular)

then

$\text{Color}_L[u_L]$ should be the same as $\text{color}_R[u_R \text{ pred from } u_L, \delta]$

- Now <u>quantize</u>: disparity to (say) r levels
- we get a discrete optimization problem in $\delta(u_L)$

$$\text{Cost}[\delta(u_L)] = \text{Loss}\left[\text{color}_L[u_L] - \text{color}_R[u_R \text{ pred from } u_L, \delta]\right]$$

- easy to rewrite as linear term using one-hot vector

Experience suggests that

$$\delta(u_i) \approx \delta(\text{neighbor of } u_i)$$

So a smoothing term would be helpful.

eg. $$\sum_{N(u_i)} \delta(u_i) \neq \delta(N(u_i))\,(\text{some weight})$$

apply one-hot, q's etc:

- same form as segmentation

- others follow this recipe (eg. denoising)

Now transform the problem yet again

$x_i \in \{1, \cdots r\}$

↑

location

so $\underline{x} \in \{1, \cdots r\}^{N_p}$

Costs:

$D_i(x_i)$ = Cost ~~when x_i~~ for x_i

$Q_{ij}(x_i, x_j)$ = Cost ~~when~~ for x_i, x_j.

have

$$\min \sum_i D_i(x_i) + \sum_i \sum_{j \in N(i)} Q_{ij}(x_i, x_j)$$

$$\text{s.t. } x_i \in \{1, \cdots r\}.$$

This isnt an lp in this form (and looks unpromising; but we had an LP, and couldn't get integer sol'n, so..)

Approximation procedure

- Start with a labelling and improve it

α - expansion :

- choose a label, α.
- every x_i with value α stays as it is
- every other x_i can either
  - stick with original label $u_i = 0$
  - change to α $u_i = 1$
- The result is a 0-1 q.p. which will be easy, under a constraint on $Q_{ij}(x_i, x_j)$

$A = \{ i \mid x_i = \alpha \}$

$B = \{ i \mid x_i \neq \alpha \}$

$$\sum_i D_i(x_i) + \sum_{i,j:\, j \in N(i)} Q_{ij}(x_i, x_j)$$

$$= \sum_{k \in A} D_k(\alpha) + \sum_{l \in B} \left[u_l \cdot D_l(\alpha) + (1 - u_l) D_l(x_l^\circ) \right]$$

$$+ \sum_{\substack{i,j \in A \times A, \\ j \in N(i)}} \left[Q_{ij}(\alpha, \alpha) \right]$$

$$+ \sum_{\substack{i \in A, l \in B, \\ l \in N(i)}} \left[Q_{il}(\alpha, \alpha) u_l + Q_{il}(\alpha, x_l^\circ)(1 - u_l) \right]$$

$$+ \sum_{\substack{l \in B, i \in A \\ i \in N(l)}} \left[Q_{li}(\alpha, \alpha) u_l + Q_{li}(x_l^\circ, \alpha)(1 - u_l) \right]$$

$$+ \sum_{\substack{l \in B, m \in B \\ m \in N(l)}} \left[\begin{array}{l} Q_{lm}(\alpha, \alpha) u_l u_m + Q_{lm}(x_l^\circ, \alpha)(1 - u_l) u_m \\ + Q_{lm}(\alpha, x_m^\circ) u_l (1 - u_m) + \\ Q_{lm}(x_l^\circ, x_m^\circ)(1 - u_l)(1 - u_m) \end{array} \right]$$

and we will min p 14 by choice of $u \in \{0,1\}^{\#B}$

- recall: question of complexity depended on Quadratic term.

here

$$Q_{\ell m}(\alpha,\alpha) + Q_{\ell m}(x_\ell^0, x_m^0) - Q_{\ell m}(x_\ell^0, \alpha) - Q_{\ell m}(\alpha, x_m^0)$$

we want this to be $-$ve.
(we are minimizing, and want to drop the constraint that forces quadratic var to be 1)

Case: assume cost has form

$$Q_{ij}(x_i, x_j) = \begin{cases} 0 & \text{agree} \\ \cancel{+ve} \; K & \text{disagree} \end{cases}$$

Then:

$$Q_{\ell m}(\alpha,\alpha) + Q_{\ell m}(x_\ell^\circ, x_m^\circ) - Q_{\ell m}(x_\ell^\circ, \alpha) - Q_{\ell m}(\alpha, x_m^\circ) < 0$$

(cause $x_\ell^\circ \neq \alpha$, $x_m^\circ \neq \alpha$, so $-k$.)

We can do more:

- recall $\triangle$ inequality

$$d(a,c) \leq d(a,b) + d(b,c).$$

We could have

$$Q_{ij}(x_i, x_j) = \left[\begin{array}{l}\text{some distance between} \\ x_i,\ x_j \text{ that obeys} \\ \triangle \text{ inequality}\end{array}\right]$$

α-β swap:

- choose two labels, α and β
- all others are fixed
- $u_i = \begin{cases} 1 & x_i \text{ goes to } \alpha \\ 0 & x_i \text{ goes to } \beta \end{cases}$
- notice this is either stick or swap
  (you can do $u_i = 1 \Rightarrow$ stick, etc, but notation gets exciting)
- get a 0-1 problem in u_i

$W = \{ i \mid x_i = \alpha \text{ OR } x_i = \beta \}$ – working

$F = \{ i \mid x_i \neq \alpha \text{ AND } x_i \neq \beta \}$ – fixed

We then get

$$\sum_{i\in F} D_i(x_i) + \sum_{j\in W}\left[u_j D_i(\alpha) + (1-u_j) D_j(\beta)\right]$$

$$+ \sum_{\substack{i\in F, j\in N(i)\\ i,j}} Q_{ij}(x_i, x_j)$$

$$+ \sum_{\substack{i\in F, j\in W\\ j\in N(i)}} \left[u_j Q_{ij}(x_i, \alpha) + (1-u_j) Q_{ij}(x_i, \beta)\right]$$

$$+ \sum_{\substack{i\in W, j\in F\\ j\in N(i)}} \left[u_i Q_{ij}(\alpha, x_j) + (1-u_i) Q_{ij}(\beta, x_j)\right]$$

$$+ \sum_{\substack{i,j\in W\\ j\in N(i)}} \left[\begin{array}{l} u_i u_j Q_{ij}(\alpha,\alpha) + (1-u_i)(1-u_j) Q_{ij}(\beta,\beta) \\ + u_i(1-u_j) Q_{ij}(\alpha,\beta) + (1-u_i) u_j Q_{ij}(\beta,\alpha) \end{array}\right]$$

quadratic term

$$u_i y_j \left[Q_{ij}(\alpha,\alpha) + Q_{ij}(\beta,\beta) - Q_{ij}(\alpha,\beta) - Q_{ij}(\beta,\alpha) \right]$$

- problem is polynomial if coefficient < 0
  (check: why < 0?)
- so $Q_{ij}(\alpha,\alpha) + Q_{ij}(\beta,\beta) < Q_{ij}(\alpha,\beta) + Q_{ij}(\beta,\alpha)$

("Better to agree than disagree")

Q is <u>metric</u> if

(a) $Q(\alpha,\beta) = 0 \iff \alpha = \beta$

(b) $Q(\alpha,\beta) = Q(\beta,\alpha) \geq 0$

(c) $Q(\alpha,\beta) \leq Q(\alpha,\gamma) + Q(\gamma,\beta)$ ($\triangle$ ineq.!)

<u>Q</u> is <u>semimetric</u> if (a), (b) ~~but~~ ~~n~~. true.

Q <u>semimetric</u> $\Rightarrow$

$$Q(\alpha,\beta) + Q(\beta,\alpha) > Q(\alpha,\alpha) + Q(\beta,\beta)$$

↑ <u>atleast</u>!

$\Rightarrow$ α - β swap OK

Q <u>metric</u> $\Rightarrow$

$$Q(l_i,\alpha) + Q(\alpha,l_j) > Q_{ij}(l_i,l_j) + Q(\alpha,\alpha)$$

↑ atleast

$\Rightarrow$ α - exp OK.

What graph should I cut?

I have

$$\sum_{ij} \left[\begin{array}{l} E_{ij}(0,0)\,(1-x_i)(1-x_j) \;+ \\ E_{ij}(1,0)\;x_i\;(1-x_j) \;+ \\ E_{ij}(0,1)\,(1-x_i)\,x_j \;+ \\ E_{ij}(1,1)\;x_i\,x_j \end{array} \right] + \sum_i \left[\begin{array}{l} E_i(0)(1-x_i) \\ +\;E_i(1)\,x_i \end{array} \right]$$

$x \in \{0,1\}$

$\rightarrow$ I need to get it onto a graph, and cut that — what graph do I cut? (earlier notes rather vague)

Strategy:
1) describe a graph rep'n for component fn's
2) Show how to add.

Graph repn:

(a) ~~Notice~~ graph has $n+2$ nodes

s, t, x_i

(b) Set up so:

$x_i \in S$ side of cut $\Rightarrow x_i = 0$

$x_i \in t$ side $\Rightarrow x_i = 1$

(c) Val(Cut) = Energy(vars) + const

Repn linear functions:

$$\sum_i \left[E_i(0)(1-x_i) + E_i(1)\, x_i \right]$$

- do one term (then sum).

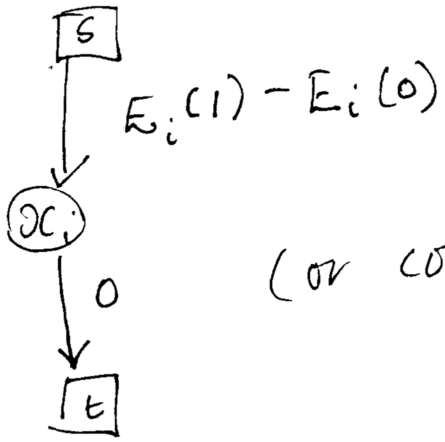
Recall capacities are ≥ 0 for max flow - mincut.

So we want a graph where
(a) ~~cap.~~ cuts rep'n cost
(b) capacities ≥ 0

__linear__

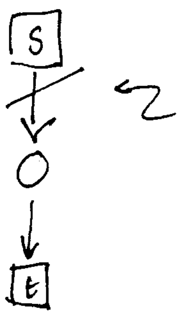
we have $E_i(1)\, x_i + (1-x_i)\, E_i(0)$

__Case 1:__ $E_i(1) > E_i(0)$.



(or could omit, which is usual)

__Notice 2 cuts :__



value is $E_i(1) - E_i(0)$

- value is 0

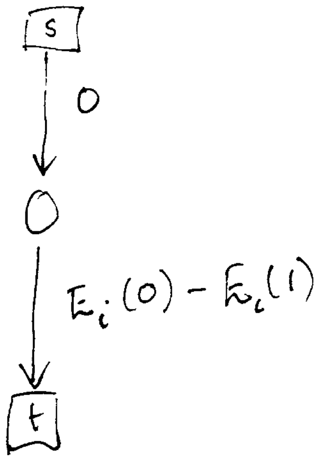
So:

Value of cut = Energy represented by cut $- E_i(0)$

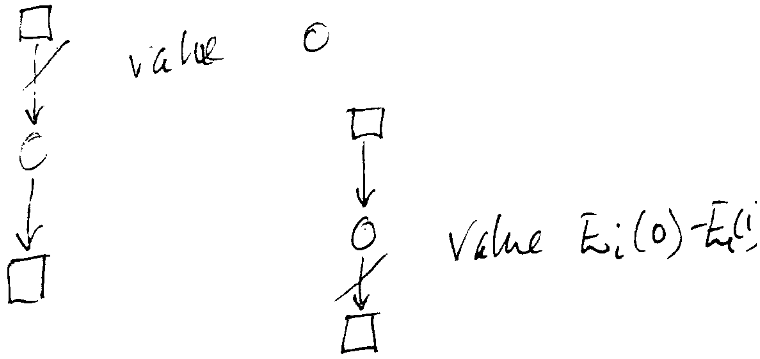
Case 2:

$E_i(1) \not> E_i(0)$

not greater than.



Again 2 cuts:



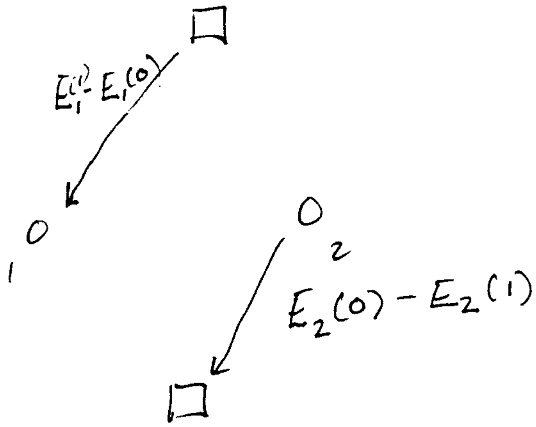
So:

value of cut = Energy repn by cut $- E_i(1)$

So now we can do any linear function.

$$E_1(0)\,\cancel{x}\,(1-x_i) + E_1(1)\,x_i + E_2(0)(1-x_i) + E_2(1)\,x_i$$

$$|||$$



(say)

etc.

Rep'n Binary (= Quadratic) terms

Notice we can decompose

$$\sum_{ij}\left[\begin{array}{l} E_{ij}(0,0)(1-x_i)(1-x_j) + E_{ij}(1,0)\,x_i\,(1-x_j) \\ + E_{ij}(0,1)(1-x_i)x_j + E_{ij}(1,1)\,x_i\,x_j \end{array}\right]$$

as a sum of terms. — we need only show how to deal w/ 1 pair

Decompose as:

constant

$E_{ij}(0,0)$.

$+ (1-x_i)\,0 \quad + x_i(E_{10}-E_{00})$

$+ (1-x_j)\,0 \quad + x_j(E_{11}-E_{10})$

linear

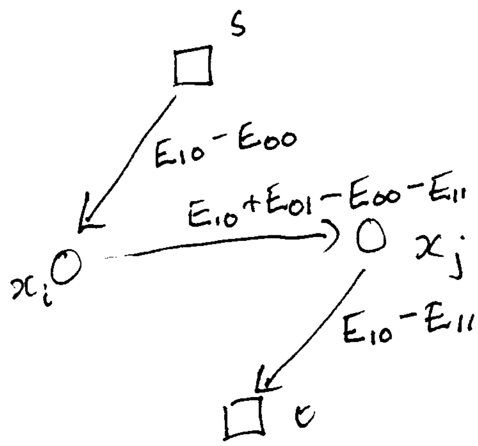
$+ (1-x_i)\,x_j\,(E_{10}+E_{01}-E_{00}-E_{11})$

interesting.

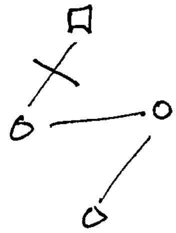
Cases depend on signs of linear coeffs:

Case 1) $\quad E_{10}-E_{00} > 0 \quad , \quad E_{10}-E_{11} > 0$

(Notice $1,0$ can't be soln)



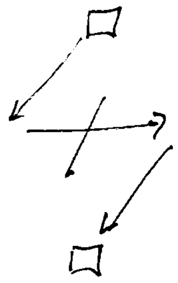
cuts



$Val(cut) = E_{10}-E_{00}$

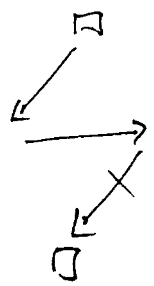
$Energy = E_{11}$

$= Val(cut) - E_{10} + E_{00} + E_{11}$



$$Val(Cut) = E_{10} + E_{01} - E_{00} - E_{11}$$

$$\begin{aligned} Energy &= E_{01} \\ &= Val(cut) - E_{10} + E_{00} + E_{11} \end{aligned}$$



$$Val(Cut) = E_{10} - E_{11}$$

$$\begin{aligned} Energy &= \cancel{E_{01}}\ E_{00} \\ &= Val(cut) - E_{10} + E_{00} + E_{11} \end{aligned}$$

So: ~~Val(Cut)~~ $Energy = Val(cut) + Const$

Case 2)

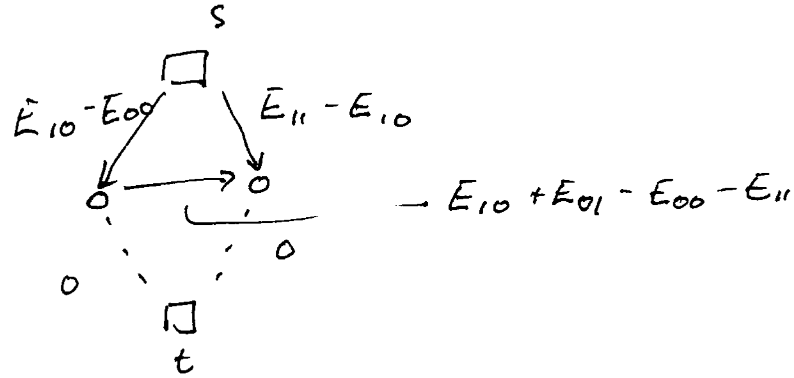
$E_{10} - E_{00} > 0$, $E_{10} - E_{11} < 0$

Notice this implies E_{00} is smallest

$$E_{11} > E_{10} > E_{00}$$

and $E_{10} + E_{01} > E_{00} + E_{11}$

$$\Rightarrow E_{01} > E_{00}$$



$Val(Cut) = \cancel{E_{00}}\ 0$

$Energy = E_{00}$

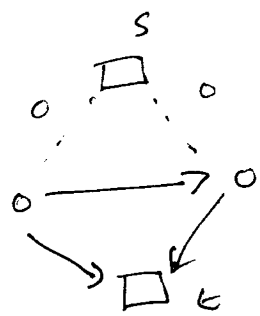
$= Val(Cut) + E_{00}$

for all cuts

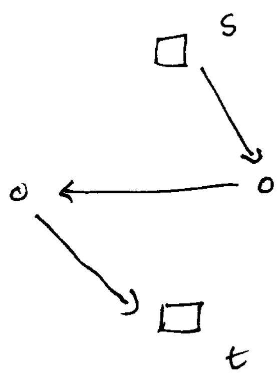
Case 3)

$E_{00} > E_{10}$

$E_{10} > E_{11}$



Case 4)



We can now do any 01
QF such that

$$E_{ij}(10) + E_{ij}(01) > E_{ij}(00) + E_{ij}(11)$$

- one node for each var
- for each linear term, insert edges, summing weights as needed
- " quad "

Val(cut) = Energy + const
=> Min (cut) gives min(Energy)

This works both ways
(i.e. represent as cut => energy cond).