6.1 NP-hard exact energy minimization

When energy functions involve multiple labels per pixel, the exact global minimization is an NP-hard problem, apart from a few special cases with very restrictive conditions on the energy functions. The pixel-wise stochastic global minimization with simulated annealing (SA) or the deterministic pixel-wise local minimization with the “greedy” iterated conditional modes (ICM) algorithm produce typically very poor results. Even in the simplest case of binary labeling considered in Lecture 5, Section 5.3, these algorithms converge to stable points that are too far from the global minimum\(^1\) \cite{14}.

The main drawback of these algorithms is that each iteration changes only one label in a single pixel in accord with the neighboring labels and therefore it results in an extremely small move in the space of possible labelings. Obviously, the convergence rate should become faster under larger moves that simultaneously change labels in a large number of pixels.

\(^1\)Although simulated annealing provably converges to the global minimum of energy \cite{11}, this could be obtained only in exponential time; this is why no practical implementation closely approaches the goal.
6.2 Approximate multiway cut and $k$-cut

As was already mentioned in Section 5.3.1, the minimum multiway cut problem of finding the minimum capacity set of edges whose removal separates a given set of $k$ terminals is NP-hard. But it has a provably good approximate solution that can be obtained with the exact min-cut / max-flow techniques. Let an isolating cut for a terminal $s_i$ in a given set $S = \{s_1, \ldots, s_k\}$ be defined as a subset of edges whose removal separates $s_i$ from the rest of the terminals. Then the following algorithm gives an approximate solution to the multicut problem [22]:

**Algorithm AMC ("Approximate Multiway Cut")**

**Step 1:** For each $i = 1, \ldots, k$, compute a maximum flow isolating cut $C_i$ by identifying the terminals in $S \setminus \{s_i\}$ into a single node and finding a minimum cut separating this node from $s_i$.

**Step 2:** Discard the heaviest of these cuts, and output the union $C$ of the rest.

Step 1 exploits $k$ separate max-flow / min-cut computations. Because the removal of $C$ from the graph disconnects every pair of terminals, it is a multiway cut.

**Theorem 6.1** (Vazirani, 2003 [22]). Algorithm AMC guarantees a solution within a factor $2 - \frac{2}{k}$ of the optimal.

**Proof.** Let $C_{\text{opt}}$ be an optimal multiway cut in a graph $G = [N,E]$. Then $C_{\text{opt}}$ can be considered as the union of $k$ cuts as follows:

- The removal of $C_{\text{opt}}$ from $G$ will create $k$ connected components, each having one terminal; since $C_{\text{opt}}$ is a minimum capacity multiway cut, no more than $k$ components will be created.

- Let $C_{\text{opt},i}$ be the cut separating the component containing $s_i$ from the rest of the graph. Then $C_{\text{opt}} = \bigcup_{i=1}^{k} C_{\text{opt},i}$.

Since each edge in $C_{\text{opt}}$ is incident at two of these components, each edge will be in two of the cuts $C_{\text{opt},i}$. Hence, $\sum_{i=1}^{k} c(C_{\text{opt},i}) = 2c(C_{\text{opt}})$.

Because $C_{\text{opt},i}$ is an isolating cut for $s_i$ and $C_i$ is a minimum capacity isolating cut for $s_i$, it holds that $c(C_i) \leq c(C_{\text{opt},i})$. This already gives a factor 2 algorithm, by taking the
union of all \( k \) cuts \( \mathcal{C}_r \). But since \( \mathcal{C} \) is obtained by discarding the heaviest of the cuts \( \mathcal{C}_i 
abla \)

\[
c(\mathcal{C}) \leq \left(1 - \frac{1}{k}\right)^k \sum_{i=1}^k c(\mathcal{C}_i) \leq \left(1 - \frac{1}{k}\right)^k \sum_{i=1}^k c(\mathcal{C}_{\text{opt},i}) = 2 \left(1 - \frac{1}{k}\right) c(\mathcal{C}_{\text{opt}})
\]

The **minimum \( k \)-cut problem** of finding a minimum-capacity set of edges whose removal leaves \( k \) connected components in a graph \( G \) has a natural factor \( 2 - \frac{2}{k} \) approximate solution: (i) starting from \( G \), compute a minimum cut in each connected component and remove the lightest one; (ii) repeat until there are \( k \) connected components.

This algorithm does guarantee a solution within the factor \( 2 - \frac{2}{k} \) of the optimal, however the proof is very complicated. There exist simpler algorithms achieving the same guaranteed approximation and having simpler proofs.

### 6.3 Large moves using min-cut / max-flow techniques

Two fast approximate algorithms for energy minimization developed in [4] improve the poor convergence of simulated annealing by replacing pixel-wise changes with specific large moves. The resulting process converges to a solution that is provably within a known factor of the global energy minimum.

The energy function to be minimized is

\[
E(x_1, \ldots, x_n) = \sum_{i=1}^n V_i(x_i) + \sum_{(i,j) \in \mathcal{N}} V_{ij}(x_i, x_j); \quad x_i \in \mathbb{L}; \quad i = 1, \ldots, n
\]  

(6.1)

where \( \mathbb{L} = \{1, 2, \ldots, L\} \) is an arbitrary finite set of labels, \( \mathcal{N} \subset \{1, \ldots, n\}^2 \) denotes a set of neighboring, or interacting pairs of pixels, \( V_i : \mathbb{L} \to \mathbb{R} \) is a pixel-wise potential function specifying pixel-wise energies in a pixel \( i \) under different labels, and \( V_{ij} : \mathbb{L}^2 \to \mathbb{R} \) is a pairwise potential function specifying pairwise interaction energies for different labels in a pair \( (i,j) \) of neighbors. The pixel-wise energies \( V_i(\ldots) \) can be arbitrary, but the pairwise interaction energies have to be either *semimetric*, i.e. satisfy the constraints

\[
V_{ij}(\alpha, \alpha) = 0; \quad V_{ij}(\alpha, \beta) = V_{ij}(\beta, \alpha) \geq 0 \quad \forall \alpha, \beta \in \mathbb{L}
\]
or **metric**, i.e. satisfy the same constraints plus the triangle inequality

\[ V_{ij}(\alpha, \beta) \leq V_{ij}(\alpha, \gamma) + V_{ij}(\gamma, \beta) \quad \forall \alpha, \beta, \gamma \in \mathbb{L} \]

Each pixel labeling \( \mathbf{x} \) with a finite set of indices \( \mathbb{L} = \{1, \ldots, L\} \) partitions the set of pixels \( \mathbf{R} = \{1, \ldots, n\} \) into \( L \) disjoint subsets \( \mathbf{R}_\lambda = \{i | i \in \mathbf{R}; \ x_i = \lambda \in \mathbb{L}\} \) (some of them may be empty) i.e. creates a partition \( \mathbf{P} = \{\mathbf{R}_\lambda : \lambda = 1, \ldots, L\} \). Each change of a labeling \( \mathbf{x} \) changes the corresponding partition \( \mathbf{P} \).

The approximate Boykov-Veksler-Zabih minimization algorithms work with any semi-metric or metric \( V_{ij} \) by using large \( \alpha-\beta \)-swap or \( \alpha \)-expansion moves respectively. The conditionally optimal moves are found with a min-cut / max-flow technique.

The **\( \alpha-\beta \)-swap** for an arbitrary pair of labels \( \alpha, \beta \in \mathbb{L} \) is a move from a partition \( \mathbf{P} \) for a current labeling \( \mathbf{x} \) to a new partition \( \mathbf{P}' \) for a new labeling \( \mathbf{x}' \) such that \( \mathbf{R}_\lambda = \mathbf{R}'_\lambda \) for any label \( \lambda \neq \alpha, \beta \). In other words, this move changes only the labels \( \alpha \) and \( \beta \) in their current region \( \mathbf{R}_{\alpha \beta} = \mathbf{R}_\alpha \cup \mathbf{R}_\beta \) whereas all the other labels in \( \mathbf{R} \setminus \mathbf{R}_{\alpha \beta} \) remain fixed. In the general case, after the \( \alpha-\beta \)-swap some pixels change their labels from \( \alpha \) to \( \beta \) and some others – from \( \beta \) to \( \alpha \). A special variant is when the label \( \alpha \) is assigned to some pixels previously labeled \( \beta \).

The **\( \alpha \)-expansion** of an arbitrary label \( \alpha \) is a move from a partition \( \mathbf{P} \) for a current labeling \( \mathbf{x} \) to a new partition \( \mathbf{P}' \) for a new labeling \( \mathbf{x}' \) such that \( \mathbf{R}_\alpha \subset \mathbf{R}'_\alpha \) and \( \mathbf{R} \setminus \mathbf{R}'_\alpha = \bigcup_{\lambda \in \mathbb{L}; \lambda \neq \alpha} \mathbf{R}'_\lambda \subset \mathbf{R} \setminus \mathbf{R}_\alpha = \bigcup_{\lambda \in \mathbb{L}; \lambda \neq \alpha} \mathbf{R} \). In other words, after this move any subset of pixels can change their labels to \( \alpha \).

**Energy minimization algorithms** The SA and ICM algorithms use **standard** pixel-wise relaxation moves changing one label each time. Such a move is both \( \alpha-\beta \)-swap and \( \alpha \)-expansion, so that these latter generalize the standard relaxation scheme. The algorithms based on these generalizations are sketched in Fig. 6.1.

An iteration at Step 2 performs \( L \) individual \( \alpha \)-expansion moves in the expansion algorithm and \( L^2 \) individual \( \alpha-\beta \)-swap moves in the swap algorithm. It is possible to prove that the minimization terminates in a finite number of iterations being of order of the image size \( n \). Actually, image segmentation and stereo reconstruction experiments in [3, 4] have shown that these algorithms converge to the local energy minimum just in a few iterations.
Swap algorithm for semimetric interaction potentials

1. **Initialization:** an arbitrary labeling $\mathbf{x}$

2. **Iterative minimization:** For every pair of labels $(\alpha, \beta) \in \mathbb{L}^2$ in a fixed or random order,
   2.1. Find $\hat{\mathbf{x}} = \arg\min_{\text{one } \alpha-\beta\text{-swap of } \mathbf{x}} E(\mathbf{x})$ with a min-cut/max-flow technique
   2.2. If $E(\hat{\mathbf{x}}) < E(\mathbf{x})$, then accept the lower-energy labeling: $\mathbf{x} \leftarrow \hat{\mathbf{x}}$

3. **Stopping rule:** If a new labeling has been accepted for at least one pair of labels at step 2.1, continue the minimization process by returning to Step 2; otherwise terminate the process and output the final labeling $\mathbf{x}$

Expansion algorithm for metric interaction potentials

1. **Initialization:** an arbitrary labeling $\mathbf{x}$

2. **Iterative minimization:** For every label $\alpha \in \mathbb{L}$ in a fixed or random order,
   2.1. Find $\hat{\mathbf{x}} = \arg\min_{\text{one } \alpha\text{-expansion of } \mathbf{x}} E(\mathbf{x})$ with a min-cut/max-flow technique
   2.2. If $E(\hat{\mathbf{x}}) < E(\mathbf{x})$, then accept the lower-energy labeling: $\mathbf{x} \leftarrow \hat{\mathbf{x}}$

3. **Stopping rule:** If a new labeling has been accepted for at least one label at Step 2.1, continue the minimization process by returning to Step 2; otherwise terminate the process and output the final labeling $\mathbf{x}$

**Figure 6.1:** Block-diagrams of the approximate minimization algorithms proposed in [4].

Given a current labeling $\mathbf{x}$ (partition $\mathbf{P}$) and a pair of labels $(\alpha, \beta)$ or a label $\alpha$, the swap or expansion move, respectively, at Step 2.1 in Fig. 6.1 use the min-cut / max-flow optimization technique to find a better labeling $\hat{\mathbf{x}}$. The latter minimizes the energy over all labelings within one $\alpha-\beta$-swap (the swap algorithm) or one $\alpha$-expansion (the expansion algorithm) of $\mathbf{x}$ and corresponds to a minimum cut of a specific graph having $O(n)$ nodes associated with pixels. The swap and expansion graphs are different, and the exact number of their pixels, their topology, and the edge weights vary from step to step in accord with the current partition.
6.3.1 Swap algorithm: finding the optimal move

The graph $G_{\alpha\beta} = [N_{\alpha\beta}, E_{\alpha\beta}]$ in Fig. 6.2 for finding an optimal swap-move is built only on the pixels $i \in R_{\alpha\beta} = R_\alpha \cup R_\beta$ having the labels $\alpha$ and $\beta$ in the partition $P$ corresponding to the current labeling $x$. The set of nodes $N_{\alpha\beta}$ includes the two terminals, denoted $\alpha$ and $\beta$, and all the pixels in $R_{\alpha\beta}$. Each pixel $i \in R_{\alpha\beta}$ is connected to the terminals $\alpha$ and $\beta$ by edges $t_{\alpha,i}$ and $t_{\beta,i}$, respectively, called $t$-links (terminal links) in [4]. Each pair of the nodes $(i,j) \subset R_{\alpha\beta}$ which are neighbors, i.e. $(i,j) \in E$, is connected with an edge $e_{i,j}$ called $n$-link (neighbor link) in [4]. Therefore, the set of edges $E_{\alpha\beta}$ consists of the $t$- and $n$-links.

![Diagram](image)

**Figure 6.2:** An example of the graph $G_{\alpha\beta}$ for finding the optimal swap move for the set of pixels $R_{\alpha\beta} = R_\alpha \cup R_\beta$ with the labels $\alpha$ and $\beta$.

If the edges have the following weights:

<table>
<thead>
<tr>
<th>Edge</th>
<th>Weight</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{\alpha,i}$</td>
<td>$V_i(\alpha) + \sum_{j \in N_i; j \notin R_{\alpha\beta}} V_{ij}(\alpha, x_j)$</td>
<td>$i \in R_{\alpha\beta}$</td>
</tr>
<tr>
<td>$t_{\beta,i}$</td>
<td>$V_i(\beta) + \sum_{j \in N_i; j \notin R_{\alpha\beta}} V_{ij}(\beta, x_j)$</td>
<td>$i \in R_{\alpha\beta}$</td>
</tr>
<tr>
<td>$e_{i,j}$</td>
<td>$V_{ij}(\alpha, \beta)$</td>
<td>$(i,j) \in \mathcal{N} \cap R_{\alpha\beta}^2$</td>
</tr>
</tbody>
</table>
then each cut \( C \) on \( G_{\alpha\beta} \) must include exactly one \( t \)-link for any pixel \( i \in R_{\alpha\beta} \); otherwise either there would be a path between the terminals if both the links are included, or a proper subset of \( C \) would become a cut if both the links are excluded. Therefore, any cut \( C \) provides a natural labeling \( x_C \) such that every pixel \( i \in R_{\alpha\beta} \) is labeled with \( \alpha \) or \( \beta \) if the cut \( C \) separates \( i \) from the terminal \( \alpha \) or \( \beta \), respectively, and the other pixels keep their initial labels (Fig. 6.3):

\[
    x_{C,i} = \begin{cases} 
    \alpha & \text{if } i \in R_{\alpha\beta} \text{ and } t_{\alpha,i} \in C \\
    \beta & \text{if } i \in R_{\alpha\beta} \text{ and } t_{\beta,i} \in C \\
    x_i & \text{if } i \notin R_{\alpha\beta}
    \end{cases}; \forall i \in R
\]

Each labeling \( x_C \) corresponding to a cut \( C \) on the graph \( G_{\alpha\beta} \) is one \( \alpha \)-\( \beta \)-swap away from the initial labeling \( x \).

![Diagram](image)

**Figure 6.3:** A cut \( C \) on \( G_{\alpha\beta} \) for two pixels \( i, j \in \mathcal{N} \) connected by an \( n \)-link \( e_{i,j} \) (dashed edges are cut by \( C \)).

Because a cut separates a subset of pixels in \( R_{\alpha\beta} \) associated with one terminal from a complementary subset of pixels associated with another terminal, it includes (i.e. severs in the graph) an \( n \)-link \( e_{i,j} \) between the neighboring pixels in \( R_{\alpha\beta} \) if and only if the pixels \( i \) and \( j \) are connected to different terminals under this cut:

- if \((t_{\alpha,i} \in C \ \text{AND} \ t_{\alpha,j} \in C) \ \text{OR} \ (t_{\beta,i} \in C \ \text{AND} \ t_{\beta,j} \in C)\) then \( e_{i,j} \notin C \)
- if \((t_{\beta,i} \in C \ \text{AND} \ t_{\alpha,j} \in C) \ \text{OR} \ (t_{\alpha,i} \in C \ \text{AND} \ t_{\beta,j} \in C)\) then \( e_{i,j} \in C \)
By taking into account that \( V_{ij}(x_c,i, x_c,j) \) is a semimetric and by considering possible cuts involving \( t \)-links of and \( n \)-link between \( i \) and \( j \) and the corresponding labelings, it is possible to prove

**Theorem 6.2** (Boykov, Veksler, Zabih, 2001 [4]). There is an one-to-one correspondence between cuts \( C \) on \( G_{\alpha \beta} \) and labelings \( x_c \) that are one \( \alpha \)-\( \beta \) swap from \( x \). The capacity of a cut \( C \) on \( G_{\alpha \beta} \) is \( c(C) = E(x_c) \) plus a constant where \( E(\ldots) \) is the energy of Eq. (6.1).

**Corollary 6.1** (Boykov, Veksler, Zabih, 2001 [4]). The lowest energy labeling within a single \( \alpha \)-\( \beta \)-swap move from a current labeling \( x \) is the labeling \( \hat{x} = x_{c^0} \) corresponding to the minimum cut \( C^0 \) on \( G_{\alpha \beta} \).

### 6.3.2 Expansion algorithm: finding the optimal move

The set of nodes \( N_\alpha \) of the graph \( G_\alpha = [N_\alpha; E_\alpha] \) for finding an optimal expansion-move (see a simple 1D example in Fig. 6.4) includes the two terminals, denoted \( \alpha \) and \( \bar{\alpha} \), all the pixels \( i \in \mathbf{R} \), and a set of auxiliary nodes for each pair of the neighboring nodes \((i, j) \in \mathcal{N}\) such that have different labels \( x_i \neq x_j \) in the current partition \( \mathbf{P} \). The auxiliary nodes are on the boundaries between the partition sets \( \mathbf{R}_\lambda; \lambda \in \mathbb{L} \). Thus the set of nodes is

\[
N_\alpha = \left\{ \alpha, \bar{\alpha}, \mathbf{R}, \bigcup_{\{i, j\} \in \mathcal{N}|x_i \neq x_j} a_{i, j} \right\}
\]

Each pixel \( i \in \mathbf{R} \) is connected to the terminals \( \alpha \) and \( \bar{\alpha} \) by \( t_\alpha,i \) and \( t_{\bar{\alpha},i} \), respectively. Each pair of the neighboring nodes \((i, j) \in \mathcal{N}\) that are not separated in the current partition, i.e. have the same labels \( x_i = x_j \) in the current labeling, is connected with an \( n \)-link \( e_{i,j} \). For each pair of the separated neighboring pixels \((i, j) \in \mathcal{N}\) such that \( x_i \neq x_j \), the introduced auxiliary node \( a_{i,j} \) results in three edges \( E_{i,j} = \{e', a_{i,j}, t_{a_{i,j}}\} \) where the first pair of \( n \)-links \( e' \) connects the pixels \( i \) and \( j \) to the auxiliary node \( a_{i,j} \), and the \( t \)-link connects the auxiliary node \( a_{i,j} \) to the terminal \( \bar{\alpha} \). Therefore, the set of all edges \( E_\alpha \) is

\[
E_\alpha = \bigcup_{i \in \mathbf{R}} \{t_\alpha,i, t_{\bar{\alpha},i}\}, \bigcup_{\{i, j\} \in \mathcal{N}|x_i \neq x_j} E_{i,j}, \bigcup_{\{i, j\} \in \mathcal{N}|x_i = x_j} e_{i,j}
\]
Figure 6.4: An example of the graph $G_\alpha$ for finding the optimal expansion move for the set of pixels in the image. Here, the set of pixels is $R = \{i, j, k, l, m\}$, and the current partition is $P = \{R_\alpha, R_\beta, R_\gamma\}$ where $R_\alpha = \{i\}$, $R_\beta = \{j, k\}$, and $R_\gamma = \{l, m\}$. Two auxiliary nodes $a_{i,j}$ and $a_{k,l}$ are added between the neighboring pixels with different labels in the current partition, i.e. at the boundaries of the subsets $R_\lambda$.

The edges have the following weights:

<table>
<thead>
<tr>
<th>Edge</th>
<th>Weight</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{\alpha,i}$</td>
<td>$\infty$</td>
<td>$i \in R_\alpha$</td>
</tr>
<tr>
<td>$t_{\alpha,i}$</td>
<td>$V_i(x_i)$</td>
<td>$i \notin R_\alpha$</td>
</tr>
<tr>
<td>$t_{\alpha,i}$</td>
<td>$V_i(\alpha)$</td>
<td>$i \in R_\alpha$</td>
</tr>
<tr>
<td>$t_{\alpha_{i,j}}$</td>
<td>$V_{ij}(x_i, x_j)$</td>
<td>$(i, j) \in N; x_i \neq x_j$</td>
</tr>
<tr>
<td>$e_{i_{a_{i,j}}}$</td>
<td>$V_{ij}(x_i, \alpha)$</td>
<td>$(i, j) \in N; x_i \neq x_j$</td>
</tr>
<tr>
<td>$e_{i_{a_{i,j}}}$</td>
<td>$V_{ij}(\alpha, x_j)$</td>
<td>$(i, j) \in N; x_i \neq x_j$</td>
</tr>
</tbody>
</table>

That any cut $C$ on $G_\alpha$ must include exactly one $t$-link for any pixel $i \in R$ provides a natural labeling $x_C$ corresponding to the cut (Fig. 6.5):

$$x_{C,i} = \begin{cases} 
    \alpha & \text{if } t_{\alpha,i} \in C \\
    x_i & \text{if } t_{\alpha,i} \notin C 
\end{cases} \quad \forall i \in R$$
Each labeling $x_c$ corresponding to a cut $C$ on the graph $G_\alpha$ is one $\alpha$-expansion away from the initial labeling $x$.

**Figure 6.5**: A minimum cut $C$ on $G_{\alpha\beta}$ for two pixels $i, j \in \mathcal{N}$ such that $x_i \neq x_j$ ($\quad \alpha \equiv \alpha_{i,j}$ is an auxiliary node between the neighboring pixels $i$ and $j$; dashed edges are cut by $C$.

Because a cut separates a subset of pixels in $\mathbf{R}$ associated with one terminal from a complementary subset of pixels associated with another terminal, it severs an $n$-link $e_{i,j}$ between the neighboring pixels $(i, j) \in \mathcal{N}$ if and only if the pixels $i$ and $j$ are connected to different terminals under this cut, or in formal terms:

\[
\begin{align*}
\text{if } t_{\alpha,i}, t_{\alpha,j} &\in C \quad \text{OR} \quad t_{\bar{\alpha},i}, t_{\bar{\alpha},j} \in C \quad \text{then} \quad e_{i,j} \notin C \\
\text{if } t_{\bar{\alpha},i}, t_{\bar{\alpha},j} &\in C \quad \text{OR} \quad t_{\alpha,i}, t_{\alpha,j} \in C \quad \text{then} \quad e_{i,j} \in C
\end{align*}
\] (6.2)

The triplet of edges $E_{i,j}$ corresponding to a pair of neighboring pixels $(i, j) \in \mathcal{N}$ such that $x_i \neq x_j$ may be cut in different ways even when the pair of severed $t$-links at $i$ and $j$ are fixed. However, a minimum cut defines uniquely the edges to sever in $E_{i,j}$ in these cases due to the minimality of the cut and the metric properties of the potentials associated with the edges $\{e_{i,\alpha_{i,j}}, e_{\alpha_{i,j},j}, t_{\alpha_{i,j}}\} \in E_{i,j}$. The triangle inequality suggests that it is always better to cut any one of them, rather than the other two together. This property of a minimum cut $C$ is illustrated in Fig. 6.5 and has the following formal representation:
if \((i,j) \in C\) and \(x_i \neq x_j\), then \(C\) satisfies the conditions
\[
\begin{align*}
\text{if } t_{\alpha,i}, t_{\alpha,j} \in C & \text{ then } C \cup E_{i,j} = \emptyset \\
\text{if } t_{\bar{\alpha},i}, t_{\bar{\alpha},j} \in C & \text{ then } C \cup E_{i,j} = t_{\bar{\alpha},i,j} \\
\text{if } t_{\bar{\alpha},i}, t_{\alpha,j} \in C & \text{ then } C \cup E_{i,j} = e_{i,a_{i,j}} \\
\text{if } t_{\alpha,i}, t_{\bar{\alpha},j} \in C & \text{ then } C \cup E_{i,j} = e_{a_{i,j}}
\end{align*}
\] (6.3)

These properties may hold for non-minimal cuts, too. If an elementary cut is defined as a cut satisfying the conditions in Eqs. (6.2) and (6.3), then it is possible to prove

**Theorem 6.3** (Boykov, Veksler, Zabih, 2001 [4]). Let a graph \(G_{\alpha}\) be constructed as above given a labeling \(x\) and \(\alpha\). Then, there is an one-to-one correspondence between elementary cuts on \(G_{\alpha}\) and labelings within one \(\alpha\)-expansion from \(x\). The capacity of any elementary cut \(C\) is \(c(C) = E(x_C)\) where \(E(\ldots)\) is the energy of Eq. 6.1.

**Corollary 6.2** (Boykov, Veksler, Zabih, 2001 [4]). The lowest energy labeling within a single \(\alpha\)-expansion move from \(x\) is the labeling \(\hat{x} = x_C^*\) corresponding to the minimum cut \(C^*\) on \(G_{\alpha}\).

### 6.4 Optimality of large moves

Although the swap move algorithm has a wider application area due to only semimetric requirements to potentials \(V_{i,j}(\ldots)\), generally it possesses no proven optimality properties. But a local minimum obtained with the expansion move algorithm is within a fixed factor of the global minimum, according to

**Theorem 6.4** (Boykov, Veksler, Zabih, 2001 [4]). Let \(\hat{x}\) be a labeling for a local energy minimum when with the expansion moves are allowed, and let \(x^*\) be the globally optimal solution. Then, \(E(\hat{x}) \leq 2\gamma E(x^*)\) where

\[
\gamma = \max_{(i,j) \in N} \left( \frac{\max_{\alpha \neq \beta \in L} V_{i,j}(\alpha, \beta)}{\min_{\alpha \neq \beta \in L} V_{i,j}(\alpha, \beta)} \right)
\]

**Proof.** Let some \(\alpha \in \mathbb{L}\) be fixed and let \(R^*_{\alpha} = \{i \in R | x^*_i = \alpha\}\). Let \(x_{\alpha}\) be a labeling within one \(\alpha\)-expansion move from \(\hat{x}\) such that

\[
x_{\alpha,i} = \begin{cases} 
\alpha & \text{if } i \in R^*_{\alpha} \\
\hat{x}_i & \text{otherwise}
\end{cases}
\]
Since $\hat{x}$ is a local minimum if expansion moves are allowed,

$$E(\hat{x}) \leq E(x_0) \quad (6.4)$$

Let $S = S_{\text{pix}} \cup S_{\text{pair}}$ be an union of an arbitrary subset $S_{\text{pix}}$ of pixels in $\mathbf{R}$; $S_{\text{pix}} \subseteq \mathbf{R}$, and of an arbitrary subset $S_{\text{pair}}$ of neighboring pixels in $\mathcal{N}$; $S_{\text{pair}} \subseteq \mathcal{N}$. A restriction of the energy of labeling $x$ to $S$ is defined as

$$E(x|S) = \sum_{i \in S_{\text{pix}}} V_i(x_i) + \sum_{(i,j) \in S_{\text{pair}}} V_{i,j}(x_i, x_j)$$

Let $I_\alpha$, $B_\alpha$, and $O_\alpha$ denote the union of pixels and pairs of neighboring pixels contained inside, on the boundary, and outside of $R_\alpha^*$, respectively:

$$I_\alpha^* = R_\alpha^* \cup \{(i,j) \in \mathcal{N} : i \in R_\alpha^* ; j \in R_\alpha^* \}$$

$$B_\alpha^* = \{(i,j) \in \mathcal{N} : i \in R_\alpha^* ; j \notin R_\alpha^* \}$$

$$O_\alpha^* = R \setminus R_\alpha^* \cup \{(i,j) \in \mathcal{N} : i \notin R_\alpha^* ; j \notin R_\alpha^* \}$$

The following three relationships hold:

(a) $E(x_0|I_\alpha^*) = E(x^*|I_\alpha^*)$; (b) $E(x_0|B_\alpha^*) \leq cE(x^*|B_\alpha^*)$; (c) $E(x_0|O_\alpha^*) = E(\hat{x}|O_\alpha^*)$

The relationships (a) and (c) follow directly from the definitions of $R_\alpha^*$ and $x_0$. The relationship (b) holds because $V_{i,j}(x_{i,j}, x_{i,j}) \leq cV(x_i^*, y_i^*) \neq 0$ for any $(i,j) \in B_\alpha^*$.

The union $I_\alpha^* \cup B_\alpha^* \cup O_\alpha^*$ includes all the pixels in $\mathbf{R}$ and all the neighboring pairs of pixels in $\mathcal{N}$. Therefore, Eq. (6.4) can be rewritten as

$$E(\hat{x}|I_\alpha^*) + E(\hat{x}|B_\alpha^*) + E(\hat{x}|O_\alpha^*) \leq E(x_0|I_\alpha^*) + E(x_0|B_\alpha^*) + E(x_0|O_\alpha^*)$$

By substituting the above relationships (a)–(c), one can obtain:

$$E(\hat{x}|I_\alpha^*) + E(\hat{x}|B_\alpha^*) \leq E(x^*|I_\alpha^*) + cE(x^*|B_\alpha^*) \quad (6.5)$$

To get the bound on the total energy, Eq. (6.5) has to be summed over all the labels $\alpha \in \mathbb{L}$:

$$\sum_{\alpha \in \mathbb{L}} (E(\hat{x}|I_\alpha^*) + E(\hat{x}|B_\alpha^*)) \leq \sum_{\alpha \in \mathbb{L}} (E(x^*|I_\alpha^*) + cE(x^*|B_\alpha^*))$$

(6.6)

For every $(i,j) \in \mathbf{B} = \bigcup_{\alpha \in \mathbb{L}} B_\alpha^*$, the term $V_{i,j}(\hat{x}_i, \hat{x}_j) = E(\hat{x}|\{(i,j)\})$ appears twice on the left side of Eq. (6.6): once in $E(\hat{x}|B_\alpha^*)$ for $\alpha = x_i^*$, and once in $E(\hat{x}|B_\alpha^*)$ for $\alpha = x_j^*$.
Similarly, every $V(x_i^*, x_j^*) = E(x^*|(i,j))$ appears $2c$ times on the right side of Eq. (6.6). Therefore, the latter equation can be rewritten as

$$E(\bar{x}) + E(\bar{x}|B) \leq E(x^*) + (2c-1)E(x^*|B) \leq 2cE(x^*)$$

to give the bound of $2c$ for the factor of the global minimum. \hfill $\square$

### 6.5 Exercises

**Exercise 6.1.** Reformulate in terms of graphs, capacity (or cost) functions on edges, neighborhood relations, and minimum cuts the general problem of minimizing an energy $E(x)$ of Eq. (6.1), i.e. finding a configuration $x^* = \arg\max_{x \in \mathbb{L}} E(x)$.

**Exercise 6.2.** Decide to what extent the reformulated energy minimization problem relates the minimum multiway cut problem and/or the minimum $k$-cut problem.

**Exercise 6.3.** Show that the graph in Fig. 6.6 presents a tight example for the AMC algorithm in Section 6.2. The edges of the cycle have capacity 1 and edges attaching terminals $s_1, s_2, s_3, s_4$ to the cycle have capacity $2 - \varepsilon$ for a small fraction $1 \gg \varepsilon > 0$.

![Figure 6.6: An example of the multiway problem with $k = 4$.](image-url)