

1 Preliminaries

- See: <http://www.mancoosi.org/misc-2012/criteria/>
- Package universe: \mathcal{P}
- Installation: $\mathcal{I} \subseteq \mathcal{P}$
- Closure: $\mathcal{C} \subseteq \mathcal{P}$
- Removed (by explicit request): $\mathcal{R} \subseteq \mathcal{P}$
- Optimization criteria: \mathcal{O}
- Names: $X.name = \{p.name \mid p \in X\}$
- Versions: $X.version(p) = \{q.version \mid q \in X, p.name = q.name\}$

2 Closure

- Table 1 lists available selectors
 - S^∞ is an upper bound for packages in a selector
 - $S^-(q)$ for $q \in S^\infty$ is the set of packages whose installation might remove q from the selector
 - $S^+(q)$ for $q \in S^\infty$ is the set of packages whose installation might add q to the selector
- a package $p \in \mathcal{P} \setminus \mathcal{R}$ is relevant if at least one of the following conditions holds
 - $+f \in \mathcal{O}$ where
 - * $f = \text{count}(S), q \in S^\infty, \text{ and } p \in S^+(q),$
 - * $f = \text{sum}(S, attr), q \in S^\infty, q.attr > 0, \text{ and } p \in S^+(q),$
 - * $f = \text{sum}(S, attr), q \in S^\infty, q.attr < 0, \text{ and } p \in S^-(q),$
 - * $f = \text{notuptodate}(S), q \in S^\infty, q.version \neq \max(\mathcal{P}.version(q)), \text{ and } p \in S^+(q),$
 - * $f = \text{unsat_recommends}(S), q \in S^\infty, q.recommends \neq \emptyset, \text{ and } p \in S^+(q), \text{ or}$
 - * $f = \text{aligned}(S, group, value), q \in S^\infty, q' \in S^\infty, q.group = q'.group, q.value \neq q'.value, \text{ and } p \in S^+(q)$
 - $-f \in \mathcal{O}$ where
 - * $f = \text{count}(S), q \in S^\infty, \text{ and } p \in S^-(q),$
 - * $f = \text{sum}(S, attr), q \in S^\infty, q.attr > 0, \text{ and } p \in S^-(q),$
 - * $f = \text{sum}(S, attr), q \in S^\infty, q.attr < 0, \text{ and } p \in S^+(q),$
 - * $f = \text{notuptodate}(S), q \in S^\infty, q.version \neq \max(\mathcal{P}.version(q)), \text{ and } p \in S^-(q),$
 - * $f = \text{unsat_recommends}(S), q \in S^\infty, q.recommends \neq \emptyset, \text{ and } p \in S^-(q),$
 - * $f = \text{unsat_recommends}(S), q \in S^\infty \text{ and } p \in q.recommends \setminus \mathcal{R}, \text{ or}$
 - * $f = \text{aligned}(S, group, value), q \in S^\infty, q' \in S^\infty, q.group = q'.group, q.value \neq q'.value, \text{ and } p \in S^-(q)$
- the closure \mathcal{C} is the least set that
 - contains all relevant packages, and
 - if $p \in \mathcal{C}$ then $p.depends \setminus \mathcal{R} \subseteq \mathcal{C}$

3 Fact Format

- `unit(p.name,p.version,in)` . for $p \in \mathcal{C}$
- `unit(p.name,p.version,out)` . for $p \in \mathcal{R}$
- `installed(p.name,p.version)` . for $p \in \mathcal{P}$
- `maxversion(p.name,p.version)` . for $p \in \mathcal{P}$ and $p.version = \max(\mathcal{P}.version(p))$

selector S	S^∞	$S^-(q)$	$S^+(q)$
solution	$\mathcal{P} \setminus \mathcal{R}$	\emptyset	$\{q\}$
changed	$\mathcal{P} \setminus \mathcal{R} \cup \mathcal{I}$	$\{q\} \cap \mathcal{I} \setminus \mathcal{R}$	$\{q\} \setminus \mathcal{I}$
new	$\{p \in \mathcal{P} \setminus \mathcal{R} \mid p.name \notin \mathcal{I}.name\}$	\emptyset	$\{q\}$
removed	\mathcal{I}	$\{p \in \mathcal{P} \setminus \mathcal{R} \mid p.name = q.name\}$	\emptyset
up	$\{p \in \mathcal{P} \setminus \mathcal{R} \mid p.name \in \mathcal{I}.name, p.version > \max(\mathcal{I}.version(p))\}$	\emptyset	$\{q\}$
down	$\{p \in \mathcal{P} \setminus \mathcal{R} \mid p.name \in \mathcal{I}.name, p.version < \max(\mathcal{I}.version(p))\}$	\emptyset	$\{q\}$
installrequest	$\{p \in \mathcal{P} \setminus \mathcal{R} \mid q \text{ mentioned in install request } p.name = q.name\}$	\emptyset	$\{q\}$
upgraderequest	$\{p \in \mathcal{P} \setminus \mathcal{R} \mid q \text{ mentioned in upgrade request } p.name = q.name\}$	\emptyset	$\{q\}$
request	$\text{installrequest}^\infty \cup \text{upgraderequest}^\infty$	\emptyset	$\{q\}$

Table 1: Selectors

- `satisfies(Name,Version,Condition)`. for $p \in \mathcal{C}$
 - applies to various parts of the CUDF-document
 - * dependencies
 - * conflicts
 - * requests
 - * recommendations
 - * keep flags
- `depends(p.name,p.version,Condition)`. for $p \in \mathcal{C}$
- `conflict(p.name,p.version,Condition)`. for $p \in \mathcal{C}$
- `recommends(p.name,p.version,Condition,Weight)`.
 - $\pm f \in \mathcal{O}$, $f = \text{unsat_recommends}(S)$, and $p \in S^\infty$
- `request(Condition)`.
 - requests
 - keep flags
- `attribute(p.name,p.version,Attribute,Value)`.
 - $\pm f \in \mathcal{O}$, $f = \text{sum}(S, attr)$, and $p \in S^\infty$, or
 - $\pm f \in \mathcal{O}$, $f = \text{aligned}(S, group, value)$, and $p \in S^\infty$, or
- `installrequest(Name,Version)`.
- `upgraderequest(Name,Version)`.
- `clique(I,p.name,p.version)` where $p \in I$ and I is a non-singular clique in the graph $(\mathcal{C}, \{(p, q) \mid p, q \in \mathcal{C}, p \neq q, q \in p.conflicts\})$
 - no overlapping cliques will be added
- `criterion(Maximize,Selector,count,Priority)`.
- `criterion(Maximize,Selector,sum(attr),Priority)`.
- `criterion(Maximize,Selector,notuptodate, Priority)`.
- `criterion(Maximize,Selector,unsat_recommends,Priority)`.
- `criterion(Maximize,Selector,aligned(group, value),Priority)`.