I.5 The *t*-strategies

The practical performances of all the Bundle algorithms are strongly dependent on the on-line choice of one critical *algorithmic parameter* (t, ε and l for ($\Delta^{p}_{\beta t}$), (M^{ε}_{β}) and ($\Lambda_{\beta l}$) respectively) whose value is admittedly hard to guess [HL93 Remark XIV.4.2.1, comments to Algorithm XV.3.3.4]. Most of the *t*-strategies proposed in the literature are essentially heuristics, as described in §I.5.1: however, more sophisticated *t*-strategies have also been devised, like the BT approach to which §I.5.2 is devoted. Unfortunately, all these *t*-strategies share a rather negative behaviour, the "small *t*" phenomenon, at least when used for polyhedral functions with many variables: this is probably due to the fact that all those approaches are "myopic" in the time dimension, and focus exclusively on the "dual" side of the problem (achieving a large value if $\varphi()$) disregarding the "primal" one (finding a null subgradient, or equivalently a feasible primal solution). Hence, in §I.5.3 two "Long-Term" *t*-strategies are proposed that try to avoid these pitfalls, and some encouraging computational results are reported.

I.5.1 Heuristic *t*-strategies

A number of heuristic rules can be proposed as *t*-strategies: surprisingly, only [Ki90] seem to have appeared in the literature. There, the idea is that of approximating the restriction of $\varphi()$ along $\boldsymbol{d}$ with a quadratic function $q()$ such that $q(0) = \varphi(\bar{\boldsymbol{y}})$, $q(t) = \varphi(\boldsymbol{y})$ and having $q'(0) = \boldsymbol{v}$, yielding

(a) $$t_a = \frac{tv}{2(v - \Delta\varphi)};$$

like all the heuristic rules, *(a)* can be questioned in several ways, the basic one being perhaps just wondering why should a quadratic function be any good in predicting the behaviour of a (say) piecewise-linear one. However, even if this is accepted, it is as well possible to rather fix $q'(t) = \boldsymbol{dg}$, yielding the alternative rule

(b) $$t_b = t\frac{2\Delta\varphi - \boldsymbol{dg}}{2(\Delta\varphi - \boldsymbol{dg})} = t\frac{\alpha + \Delta\varphi}{2\alpha}$$

that has the nice property that $t_b > t \Leftrightarrow \boldsymbol{dg} > 0$ (see Figure 21 and Figure 22); another possibility could be to maximize the piecewise linear function constructed by the two linearizations at 0 and $\boldsymbol{y}$ (restricted to $\boldsymbol{d}$), yielding

(c) $$t_c = \frac{\alpha t - 0}{v - \boldsymbol{d}\,\boldsymbol{g}}.$$

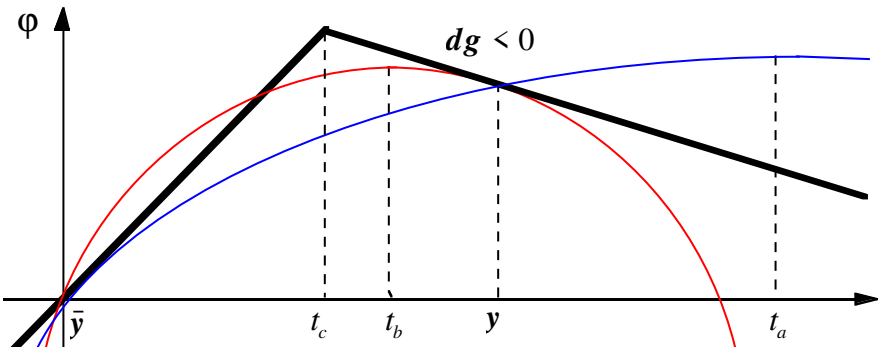

Figure 21: rules *(a)*, *(b)* and (*c*) can give different values of t for $\boldsymbol{dg} < 0$

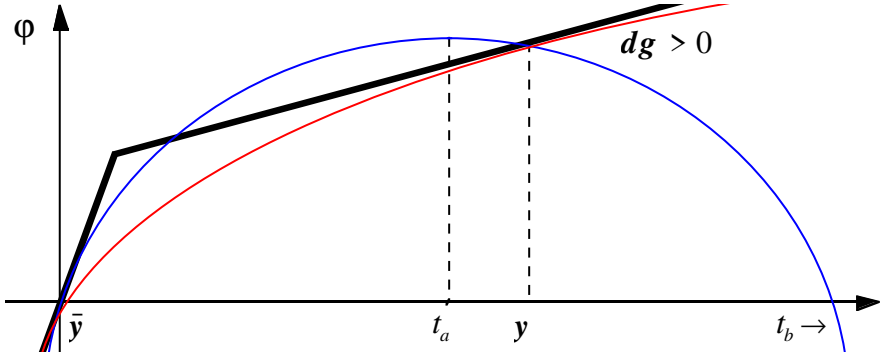

Figure 22: rules *(a)* and *(b)* can give different values of t for $\boldsymbol{dg} > 0$

Each of the above formulas have a "degenerate" case where it is not capable of giving sensible results (respectively $v \approx \Delta\varphi$, $\alpha \approx 0$ and $v \approx \boldsymbol{dg}$), but a more relevant issue of the approach is that only information relative to the current iteration is used: the method can be extended to have some sort of "memory", as in [Ki90] where t is increased or decreased - using *(a)* - only after *three* consecutive SS or NS, but this does not appear to be a really satisfactory solution.

However, the above formulas do *not depend on the choice of p*, i.e. they keep their heuristic validity - if any - in the $p \in \{1, \infty\}$ cases also: surprisingly, this is (almost) true even for the most sophisticated t-strategy proposed so far, the Bundle / Trust-Region method described in the next paragraph.

I.5.2 The Bundle / Trust-Region algorithm

The *Bundle / Trust-Region* (BT) algorithm, introduced in [SZ92], is described in Figure 23: in the pseudo-code, C_1 and C_2 stand for extra conditions that will be discussed later on. The main ideas of the algorithm are standard: in order to accept the new $\boldsymbol{y}$ as the current point, the *obtained* increase $\Delta\varphi$ must be at least a constant fraction (m_1) of the *predicted* increase $v = \varphi_\beta(\boldsymbol{d})$ (condition SS(i)). Measuring *relative* improvements, the SS(i) condition depends on how "demanding" one had been at the previous iteration, i.e. on t, and has a potential drawback in that the *absolute* magnitude of $\Delta\varphi$ is completely disregarded, possibly leading to situations where a negligible improvement is considered important. Condition SS(*ii*) tries to avoid this problem by detecting the case in which a (much) better improvement could have been obtained by augmenting t: since moving to $\boldsymbol{y}$ would nullify the linearization error of newly obtained $\boldsymbol{g}$ ($\in \partial\varphi(\boldsymbol{y})$), the condition $\boldsymbol{dg} + 0 \le m_2 v < v$ ensures that the current $(v, \boldsymbol{d})$ will no more be optimal to ($\Pi_{\beta'}$) if $\boldsymbol{g}$ is added to β, i.e. that the new information has

seriously modified the Cutting Plane model.

Bundle/Trust Region Algorithm

⟨ choose $0 < m_1 < m_2 < 1$, $0 < m_3 < 1$, $t^* > 0$, $\varepsilon > 0$, the initial $\bar{y}$ and t ⟩; $\beta = \{ g(\bar{y}) \}$;
do
⟨ solve $(\Pi^p_{\beta t})$ and $(\Delta^p_{\beta t})$ for (v, d) and $(\sigma, \|z\|_p)$ ⟩;
$y = \bar{y} + d$; $g = g(y)$; $\Delta\varphi = \varphi(y) - \varphi(\bar{y})$; $\alpha = \Delta\varphi - dg$;
if($\Delta\varphi \geq m_1 v$) /* SS(*i*) */
then **if**(($dg \leq m_2 v$) & ~C_1) /* SS(*ii*) */
then $\beta = \beta \cup \{ g \}$; $\bar{y} = y$;
else Increase(t);
else /* NS(*i*) = ~SS(*i*) */
if(($\alpha \leq m_3 \sigma$) & C_2) /* NS(*ii*) */
then $\beta = \beta \cup \{ g \}$;
else Decrease(t);
while($\sigma + t^* \|z\|_p > \varepsilon$);

Figure 23:the Bundle/Trust Region algorithm

However, dg is also the slope of $\varphi_\beta()$ in d, i.e. how much could be gained - at most - by moving to $2d$ instead of d (see Figure 24): hence, ~SS(*ii*) is true if the maximum improvement that might be obtained by doubling t (if d would remain the same *and* $\varphi_\beta()$ would be good approximation of $\varphi()$) is a significative fraction of v; condition C_1 is just "$t \approx \bar{t}$", i.e. the SS has to be done anyway if t is already (almost) as large as it can be. The NS(*ii*) condition is quite clear: the newly obtained subgradient g is trusted as a relevant first-order information about $\varphi()$ in $\bar{y}$ only if it is of about the same "quality" of z, i.e. if it belongs to the $(m_3\sigma)$-subdifferential of $\varphi()$ in $\bar{y}$, m_3 being another fixed constant. Actually, the idea is that m_3 should be smaller than 1, since z is "poor" (a NS has just been done) and therefore something better than z is needed: if it is not available, a t decrease is preferred.

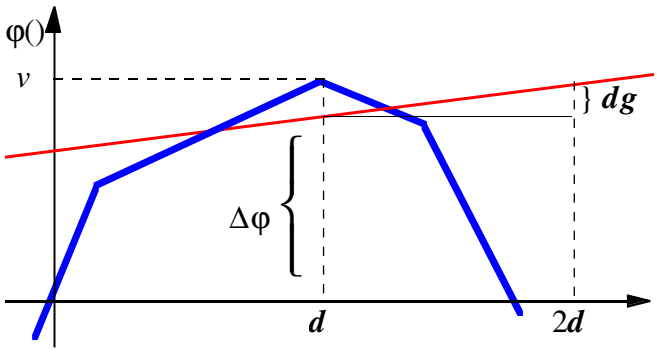

Figure 24: the SS(*ii*) rule and the expected improvement in $\bar{y} + 2d$

However, the interesting point is that the overall algorithm is constructed in such a way that a *Curve-Search* procedure on the y space is performed for seeking the "best" value of t: this is based on the fact that, if SS(*i*) and ~SS(*ii*) are true for a given t, then there exists a $t' > t$ for which both SS(*i*) and SS(*ii*) will be true, and the same holds for the NS() conditions (note that the new g is *not* inserted in β in these cases just for ensuring this property, and that the last part of our condition *(v)* exactly accounts for this extra degree of freedom). Hence, the algorithm maintains a "feasible interval" $[l , u] \subseteq [0 , \bar{t}]$ containing t, and the IncreaseT(), DecreaseT() procedures respectively set $l = t$, $u = t$ and choose the new t in the shrunken interval: some safeguards are necessary, in theory, to guarantee that the interval will eventually become smaller than any fixed threshold in a finite number of steps, and in

fact a simple bisection method is analyzed in [SZ92], but some of the heuristics of §I.5.1 could be used as well. The C_2 condition is just "$l = 0$", telling that the algorithm is performing a Curve Search where SS(i) and ~SS(ii) have been true at least once, so that only a SS will be accepted as its final result (l and u are then reset to $\underline{t}$, $\bar{t}$): in fact, it can be shown that (for $p = 2$) the CS ends in a finite number of steps with either a SS or a NS. Note that this is *not* done to maximize $\varphi()$ - the final SS might well *not* lead to the best point found during the CS - but rather to find a "good" value of t by sampling the space around $\bar{\boldsymbol{y}}$: this is done by letting $\boldsymbol{d}$ change in response to the t variation, although it may be also reasonable to do it along a fixed direction ("resuscitating" the Line Search for a different purpose) if the solution of ($\Delta^p_{\beta t}$) would be the most time-consuming part of the whole approach.

Having been proposed for the $p = 2$ case, the original BT algorithm does not fit exactly in the convergence proofs of Chapter I.2 and §I.4.2: in particular, it does not use a lower bound $\underline{t}$ on t, hence *(iii)* does not necessarily holds, and furthermore the proof that the CS is finite uses the *unicity* of the optimal solution of ($\Pi^2_{\beta t}$), that does not extend to the $p \in \{1, \infty\}$ cases. However, it is easy to see that only minor changes are needed in order to fit it into the new setting: for instance, finite termination of the CS can be also guaranteed by simple conditions like $u - l < \zeta$ (that are in practice advised anyway, and with a not too small $\zeta > 0$). Furthermore, the fact that a SS is always performed at the end of a CS where SS(i) has been true at least once is not needed for convergence, but can still be easily ensured with the simple device of saving the t (and $\boldsymbol{y}$, v ...) of the latest such iteration and restoring it if the CS is stopped at a NS, obtaining an essentially equivalent algorithm. These changes can be easily made in such a way that t is always kept larger than $\underline{t}$, so that (due to the bisection mechanism, or the equivalent safeguards) *(i)*, *(iii)* and *(iv)* are restored in the same time: since SS(i) is just condition *(ii)*, and *(v)* is automatic, BT can be extended to the new class without loosing any of its main characteristics.

I.5.3 Long Term *t*-strategies

In practice, the behaviour of Bundle algorithms on large-scale NDO problems is sometimes not satisfactory: in our experience, all the above t-strategies work well only during an initial phase, while afterwards t rapidly decreases, since the conditions required for a t augment are almost never satisfied. As a consequence, a long sequence of very "short" Serious Steps is performed, resulting in extremely slow convergence: usually, by just keeping t fixed to some "large" value (that, however, must be choosen with a rule of thumb) the algorithms perform much better. Some "philosophical" observations can be made about these difficulties:

- all the above t-strategies lack of any (at least tentative) *long term memory*: only the outcome of the last (few) iteration(s) is used to choose t, so that - even tough the information about $\varphi()$ is kept - Bundle methods suffer from a lack of information about their own past choices;
- all the above t-strategies measure *relative increments*, and the absolute magnitude of the obtainable increments is completely disregarded;
- in all cases, the "ideal" t that is searched for should permit a sequence of SS only, giving a measure of the "curvature" of $\varphi()$, i.e. how "far" it is possible to go before the function "changes too

much": for polyhedral functions arising from Lagrangean relaxations this may not be a wise choice, as they have an enormous number of facets, so that such a t is likely to be that very small;

– finally, none of the above t-strategies takes into account the reduction of $\| z \|_p$, i.e. the "primal side" of the problem, although a small value of the norm is required for termination.

In this paragraph, two new "Long Term" t-strategies are proposed that try to deal with the above problems: from the dual viewpoint, they are essentially based on keeping a *measure* of what a *sensible improvement* is, and using this information for increasing t (or inhibiting the decreases) if necessary; from the primal viewpoint, they can be seen as Dantzig-Wolfe approaches with *repeated but approximated* Phases 0. These strategies - that are *independent* from the choice of p - are not designed to *replace* the above (heuristic or not) t-strategies, but rather to *integrate* them: they should be applicable to almost every known Bundle algorithm, such as BT of §I.5.2, with only minor modifications in its original rules.

The "Hard" version, described in Figure 25, keeps an "expected minimum increase" $\bar{\varepsilon}$, which is the "Long Term memory" of the method: a step to $\boldsymbol{y} = \bar{\boldsymbol{y}} + \boldsymbol{d}$ is considered useless if the maximum obtainable improvement, measured by v, is not large enough, i.e. greater than a constant fraction $m_4 \leq 1$ of $\bar{\varepsilon}$. In this case, enlarging t is preferred: the new t is chosen by using *sensitivity analysis* information about the $(\Delta^p_{\beta t})$ subproblem, i.e. the two numbers $s \geq 0$ and q (Cf. §I.3.6 and §I.1.2 for $p = 2$) such that $ts + q$ is an upper bound on $v(t)$, and it is set in such a way that $v \geq \bar{\varepsilon}$ at the next iteration. Since $ts + q$ is only an upper bound on $v(t)$, more than one US (but only finitely many) may be required to find the correct value of t.

Hard Long Term t-strategy

⟨ choose $t^* > 0$, $\bar{\varepsilon} \geq \varepsilon > 0$, the initial $\bar{\boldsymbol{y}}$, t and β ⟩;
do
  ⟨ solve $(\Pi^p_{\beta t})$ and $(\Delta^p_{\beta t})$ for $(v, \boldsymbol{d})$ and $(\sigma, \| z \|_p)$ ⟩;
  if($\sigma + t^* \| z \|_p \leq \bar{\varepsilon}$) /* convergence for $\bar{\varepsilon}$ */
    $\bar{\varepsilon} = (\sigma + t^* \| z \|_p) / 2$;
  if(($v < m_4 \bar{\varepsilon}$) & ($t < t^*$)) /* Useless Step */
    $t = \min\{ t^* , (\bar{\varepsilon} - q) / s \}$;
  else /* a "normal" step */
    ⟨ any Bundle algorithm, eventually modified to
      "avoid interference" with the US mechanism ⟩;
while($\sigma + t^* \| z \|_p > \varepsilon$);

Figure 25: the "hard" Long Term t-strategy

$\bar{\varepsilon}$ is decreased once no other increments of the current magnitude are expected, i.e. when *an $\bar{\varepsilon}$-optimal solution is obtained* (assuming that t^* has been correctly chosen): note that $s = 0$ ($\Rightarrow (\bar{\varepsilon} - q) / s = \infty$) implies that the CP model is bounded and the current $\bar{\boldsymbol{y}}$ is q-optimal, so that setting $t = t^*$ will force "convergence" at the next step. In this case, our new stopping rule is clearly benefical, since the old "$\| z \|^2 \leq \varepsilon_Q$ & $\sigma \leq \varepsilon_L$" one would have required two different $\bar{\varepsilon}_Q$ and $\bar{\varepsilon}_L$; needless to say, this approach extends to the use of a *relative* $\bar{\varepsilon}$ as outlined at the end of §I.1.1. Note that s is smaller than the largest

directional derivative of $\varphi_\beta()$ in a properly defined ball B (of "radius" $\bar{t}$) around $\bar{\boldsymbol{y}}$, hence $s \le L$ for some Lipschitz constant $L > 0$: therefore, a lower bound exists on the t increment of a US, since $v = ts + q < m_4\bar{\varepsilon}$ and $t' = (\bar{\varepsilon} - q) / s$, so that

$$t' - t > (\bar{\varepsilon} - q) / s - (m_4\bar{\varepsilon} - q) / s = (1 - m_4)(\bar{\varepsilon} / s) \ge (1 - m_4)(\bar{\varepsilon} / L).$$

When $v \ge m_4\bar{\varepsilon}$, a "normal" Bundle algorithm is used, possibly modified for ensuring that conditions *(i)* - *(v)* are respected: usually, only minor changes are needed, as testified by the case of the BT approach, that has the most sophisticated t-strategy in the literature. For the purpose of the convergence proof, an iteration comprises a "normal" step and all the following consecutive USs (that are nothing but a t-strategy): it is easy to see that the only critical point is the analysis of an infinite sequence of NS, since conditions *(i)*, *(ii)*, *(iv)* and *(v)* are clearly not harmed. The US mechanism does not interfere with a CS started by an SS(*i*) condition: in fact, no US can be performed after an iteration h where SS(*i*) holds, until the CS ends, since β is not changed, $v_h \ge m_4\bar{\varepsilon}$ and $t \ge t_h$ due to the bisection mechanism, that is reset only after the SS. Furthermore, the standard BT scheme can be easily modified in such a way that *after an US,* NS(*i*) *and* ~NS(*ii*) *does not start a CS until* $\bar{\varepsilon}$ *is decreased*: t decreases are *not* totally inhibited, since a CS is still started if SS(*i*) and ~SS(*ii*) hold, but any such CS ends in a SS, so that within any infinite sequence of NS t is never decreased. For *(iii)* to hold, t must also be increased only finitely many times: however, this is clearly true since t is upper bounded and it increases of at least a fixed quantity at each US (since $\bar{\boldsymbol{y}}$ is not changing).

This t-strategy is a refined version of the "dual/primal" scheme at the end of §I.2.3, where convergence is reached for the "real" ε, t is all of a sudden increased to t^* and a sequence of NS is started: from the primal viewpoint, a Bundle algorithm using the "Hard" t-strategy is a DW approach with *partial and repeated* Phases 0, as convergence with a given $\bar{\varepsilon}$ requires $\| \boldsymbol{b} - \boldsymbol{A}\bar{\boldsymbol{x}} \|_p \le \bar{\varepsilon} / t^*$. In fact, in practice this t-strategy is "inactive" as long as "good" improvements are obtained, and then a sequence of NS + US is performed until the "convergence" is attained: the "t increasing phase" can occasionally be interrupted by a SS, but it usually continues immediately after. The computational results presented below show that this t-strategy can help in correctly sizing t, avoiding the "short steps" phenomenon: however, as $\bar{\varepsilon}$ decreases the "t increasing phases" tend to be longer and longer, and during such phases only a little progress (in terms of $\varphi()$ value) is made towards the optimum.

Although remarkably better than no Long Term strategy at all, the "Hard" t-strategy still seems to be ameliorable: in view of the above observations, a "Soft" approach is proposed in Figure 26 that, rather than forcing t to augment, just inhibits t decreases whenever v is "too small".

Soft Long Term *t*-strategy

⟨ choose $t^* > 0$, $\varepsilon > 0$, the initial $\bar{\boldsymbol{y}}$, t and β ⟩; $\bar{\varepsilon} = \infty$;
do
  ⟨ solve $(\Pi^p_{\beta t})$ and $(\Delta^p_{\beta t})$ for $(v, \boldsymbol{d})$ and $(\sigma, \| z \|_p)$ ⟩;
  $\bar{\varepsilon} = \min(\bar{\varepsilon} , (\sigma + t^*\| z \|_p) / 2)$;
  ⟨ any Bundle algorithm, where t decreases are "substantially" inhibited if $v < \bar{\varepsilon}$ ⟩;
while($\sigma + t^*\| z \|_p > \varepsilon$);

Figure 26: the "soft" Long Term t-strategy

Still, the exact details shall be dependent on the "basic" Bundle algorithm: for BT, it is only necessary to change the NS(*ii*) condition into "($\alpha \le m_3\sigma \vee v \ge \bar{\varepsilon}$) & $l = \underline{t}$", allowing t to decrease during any CS where the SS(*i*) condition has been true in at least one iteration, but avoiding t decreases if v is "small" and a t increase has not been performed previously; since *(iii)* is guaranteed by the standard BT algorithm, there is in fact nothing to do to prove convergence of this variant.

Both these *t*-strategies have some possible pitfalls: for instance, they probably depend onto a "good" choice of t^* for being effective in practice; furthermore, the Hard *t*-strategy is also dependent on the initial choice of $\bar{\varepsilon}$, and the computational experience suggests that the US mechanism should not come into play too early, a reasonable choice for $\bar{\varepsilon}$ being some small multiple of ε. However, t^* has to be guessed only within an order of magnitude, and, being in the same "measure units" of t, $\underline{t}$ and $\bar{t}$, it is usually easy to guess by looking at the t values in some trial runs; furthermore, it seems to depend only on the "scaling" of the function, i.e. to be quite constant on instances of the same class.

The following computational results display the effect of the Long Term strategies in the context of a real-word application to the Reserve Problem in telecommunications of §II.5.2: the "basic" Bundle algorithm uses $p = 2$ and heuristic *t*-strategies based on the *(a)* and *(b)* formulas, but it has shown to be computationally equivalent to BT. Table 5 reports the results for 9 instances, where v is the number of Lagrangean variables and *iter*, *time* are respectively the number of ($\Delta^p_{\beta t}$) solved and the total time taken by the algorithm to converge (with a relative $\varepsilon = 10^{-4}$): each problem has been solved with no Long Term *t*-strategy (*N*), with the Soft one (*S*) and with the Hard one (*H*), but for all the rest the code and the algorithmic parameters (m_1, m_3, t^* ...) were exactly the same. In all but one problem, the algorithm without any Long Term strategy was not able to converge within the prescribed accuracy in the maximum fixed number of iterations (1000 for the 6 smaller instances, 2000 for the 3 larger ones): for all the problems, (S) required less iterations than (H), but when using (H) the number of φ() evaluations may be smaller than the number of ($\Delta^p_{\beta t}$) solved, and in fact the running time of (H) is sometimes smaller than that of (S).

	N		S		H	
v	*iter*	*time*	*iter*	*time*	*iter*	*time*
783	1000	38	294	22	471	28
812	1000	83	457	36	443	25
783	1000	34	311	21	595	26
825	1000	90	761	77	796	51
644	1000	30	434	24	950	47
810	1000	23	270	15	480	16
3844	2000	1230	520	412	660	428
3844	861	615	257	314	355	254
3844	2000	774	443	352	788	681

Table 5: results for 9 real-world problems

A picture of the three variants is given in Chart 7 and Chart 8, where the { φ($\bar{y}_i$) } and { t_i } sequences (respectively) are compared for the three methods when applied to the last instance: since the variants behave almost identically for the first iterations, where they obtain a large part of the total φ() increase, the charts start from iteration 30 in order to make the "long term" behaviour more apparent. Comparing the charts, it is easy to see that a larger value of t usually corresponds to a faster convergence; on the other hand, during the "t increasing sequences" of the Hard *t*-strategy (clearly visible in Chart 8, especially the second one that reaches "convergence" near to iteration 300) the curve of Chart 7 is rather "flat", i.e. there is little (if any) progress towards the optimum. The heuristic *t*-strategies alone are not able to avoid t becoming very small (here, $\underline{t} = 10^{-3}$), so that the algorithm essentially ceases to obtain any significative improvement, although it "believes" to be doing well: on

the contrary, both the Hard and the Soft Long Term strategies are capable of keeping t large enough. For these instances, the Soft t-strategy seems to be preferable to the Hard one: heuristically, this can be justified by the fact that the "long term memory" $\bar{\varepsilon}$ in Hard t-strategy is set in a much more "exact" way than in the Soft one, thus requiring more resources that could be better used otherwise.

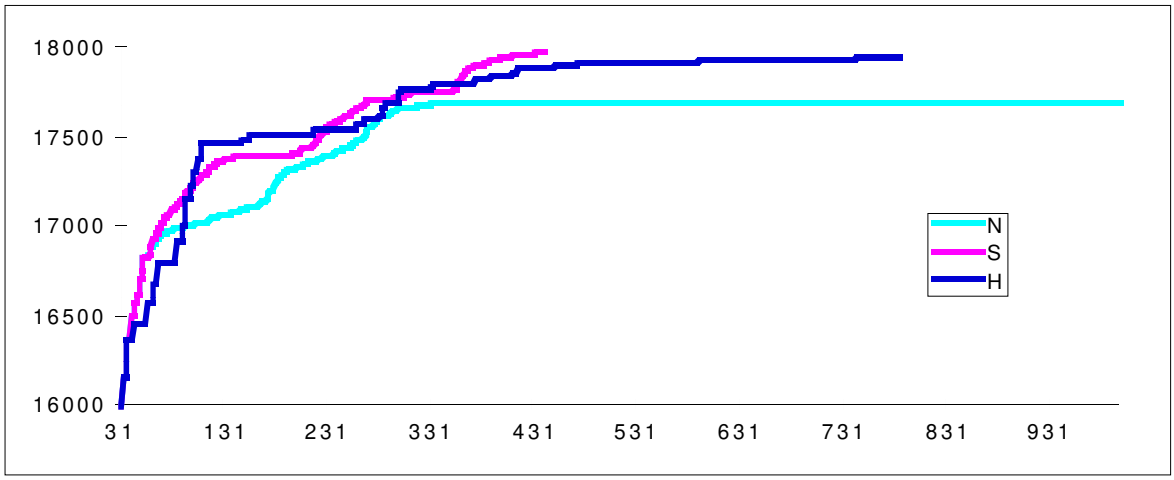

Chart 7: evolution of the value of $\varphi(\bar{\mathbf{y}}_i)$ for the three Long Term t-strategies

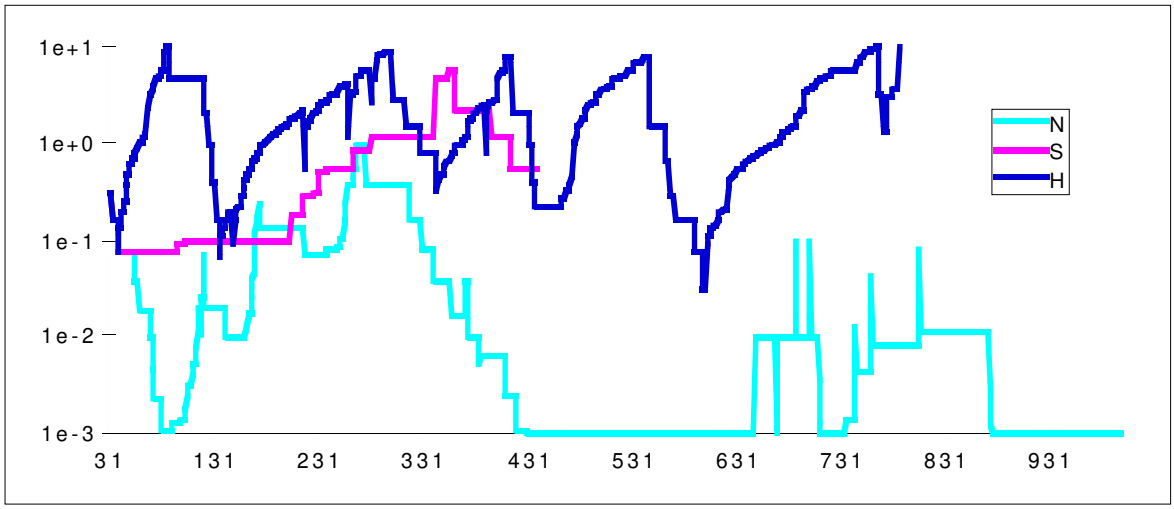

Chart 8: evolution of the value of t_i for the three Long Term t-strategies

From these results, the two Long Term strategies seem to be promising: the choice between the two - or further variants - should be dictated by the characteristics of the problem to be solved, typically the relative cost of the $\varphi()$ calculation and the $(\Delta^p_{\beta t})$ solution. Still, none of the above t-strategies is close to being "optimal", i.e. achieving a "stable" speed of convergence: however, they seem to obtain a significant improvement on the heuristic rules that have been used so far, hence there is a reasonable hope that further advances will be made in the future.

All the t-strategies presented in this Chapter have been initially proposed for the $p = 2$ case, but can be applied to the $p \in \{1, \infty\}$ cases as well: if, as it appears likely, their effectiveness is conserved in the new context, the way is open to the development of codes that exploit the efficient (LP) software currently available to diminish the cost of the "core" subproblem solution, while retaining the good behaviour of the current "quadratic" Bundle implementations.